VISIT…

LANZAROTE
Caliente.com

22.183.4
517185
3-78

БИБЛИОТЕКА СБОРНИКА
МАТЕМАТИКА

Г. ЗОЙТЕНДЕЙК

МЕТОДЫ ВОЗМОЖНЫХ НАПРАВЛЕНИЙ

METHODS OF FEASIBLE DIRECTIONS

A study in linear and non-linear programming

by

G. ZOUTENDIJK

Research Mathematician,
Koninklijke Shell Laboratories, Amsterdam

ELSEVIER PUBLISHING COMPANY

Amsterdam — London — New York — Princeton

1960

378
3-78

БИБЛИОТЕКА СБОРНИКА „МАТЕМАТИКА“

Г. ЗОЙТЕНДЕЙК

МЕТОДЫ ВОЗМОЖНЫХ НАПРАВЛЕНИЙ

Перевод с английского
С. М. МОВШОВИЧА

Под редакцией
Д. Б. ЮДИНА

ЖИТ...
ОБЛ. ...
Читальный зал № 406949

ИЗДАТЕЛЬСТВО
ИНОСТРАННОЙ ЛИТЕРАТУРЫ

Москва 1963

АННОТАЦИЯ

Автор предлагает новый метод решения задач на условный экстремум, основанный на систематическом применении метода наискорейшего спуска. Этот метод дает единообразный и систематический подход к задачам линейного, квадратичного и выпуклого программирования. Кроме того, в книге систематизируются известные методы математического программирования и указываются их видоизменения, позволяющие лучше приспособить схемы вычислений к возможностям современных вычислительных машин.

Книга написана лаконично и достаточно строго. Она представляет несомненный интерес для математиков и других специалистов, связанных с различными аспектами математического программирования.

Редакция литературы
по математическим вопросам

ПРЕДИСЛОВИЕ РЕДАКТОРА ПЕРЕВОДА

Монография Г. Зойтендейка „Методы возможных направлений" посвящена важным разделам математического программирования. Книга отличается от других работ по методам анализа задач на условный экстремум единым подходом к решению задач линейного, квадратичного и выпуклого программирования.

Монография состоит из трех частей. В первой части (гл. 1, 2) содержится конспективное изложение современного состояния математического программирования. Вторая часть (гл. 3—6) посвящена проблемам линейного программирования. Здесь систематизируются известные методы и указываются их возможные модификации, позволяющие лучше приспособить схемы вычислений к особенностям современных вычислительных машин.

Наиболее интересным разделом монографии является ее третья часть (гл. 7—11), посвященная собственно методам возможных направлений.

Методы возможных направлений, предлагаемые автором,— это итерационные методы вычисления максимума вогнутой функции на выпуклом множестве. Последовательность расчетов, определяемая методом возможных направлений, такова. Вначале вычисляется исходный план (точка области определения максимизируемой функции). Для этого в большинстве случаев требуется конечное число операций. Затем одним из указанных автором способов выбирается так называемое возможное направление. Вдоль возможного направления можно сделать шаг конечной длины, не выходя за область определения целевой функции, и увеличить при этом ее значение. Следующие итерации совершаются по тем же правилам. В задачах линейного и квадратичного программирования решение (если оно существует) достигается за конечное число шагов.

Для общей задачи выпуклого программирования гарантируется только сходимость процесса.

Методы возможных направлений представляют собо реализации метода наискорейшего спуска с конечной длино шага. Работа Зойтендейка является, по-видимому, первы систематическим применением метода наискорейшего спус к задачам на условный экстремум.

Автор показывает, что многие известные методы линей ного, квадратичного и выпуклого программирования так ил иначе связаны с идеей метода возможных направлений.

Монография написана весьма компактно, на строгом ма тематическом языке. Поэтому ее вряд ли можно рекомендо вать для первого знакомства с математическим программи рованием. Однако единый подход к разным проблема и методам, оригинальность замысла и отраженный в книг опыт практических расчетов представляют безусловный ин терес для разработки и приложения методов математическог программирования.

Д. Б. Юдин

Глава 1

ВВЕДЕНИЕ

1.1. История.

В математическом программировании изучается задача отыскания наибольших и наименьших значений функции на множестве переменных, удовлетворяющих некоторым ограничениям. Интерес к этой области математики возник в экономике и в теории управления, где многие проблемы оптимального распределения ограниченных ресурсов формулируются как задачи математического программирования. Разработка эффективных математических методов и вычислительных алгоритмов, широкое распространение быстродействующих электронных вычислительных машин дают возможность в принципе находить численное решение подобных задач. Новые методы не могут быть непосредственно выведены из классических, таких, как метод множителей Лагранжа. Метод Лагранжа эффективен в экстремальных задачах, когда ограничениями на переменные являются уравнения, но его трудно распространить на задачи с ограничениями в виде неравенств. К тому же задачи математического программирования почти всегда содержат много переменных и ограничений (их может быть несколько сотен и более). Ограничения в виде неравенств и большие размеры отличают задачи математического программирования от классических и делают необходимой разработку специальных вычислительных методов.

Крупным вкладом в эту область явилась разработка Г. Данцигом[1]) симплексного метода решения задачи линейного программирования, т. е. задачи оптимизации линейной функции на множестве переменных, удовлетворяющих системе линейных уравнений и неравенств. Целый ряд

[1]) Первые постановки и принципы решения задач линейного программирования сформулированы советским математиком Л. В. Канторовичем. — *Прим. ред.*

промышленных и коммерческих задач можно свести к
нейному программированию линеаризацией оптимизируем
функции или некоторых ограничений.

Рилеем и Гассом [27] [1]) составлена библиография по лине
ному программированию, охватывающая сотни исследован
в этой области. Линейное программирование обычно сп
собствует лучшему пониманию и решению производствен
и экономических задач. Для решения задач линейного пр
граммирования повсеместно разработаны вычислительн
программы для электронных цифровых машин.

Уделяется внимание и задачам линейного программиров
ния специального вида, таким, как транспортная или общ
задача определения максимального потока через сеть.
курсу линейного программирования написано несколько рук
водств. Математические и вычислительные аспекты теор
освещены, например, в [10, 8, 41].

Однако многие практические задачи либо совсем не св
дятся к линейным, либо сводятся к ним с большим трудо
Поэтому делалось и делается много попыток развить бол
общие методы математического программирования. Не пр
тендующий на полноту обзор существующих методов дает
в следующем пункте.

1.2. Современное состояние математического программирования.

В математическом программировании можно выделит
следующие три направления:

1) прикладные или технологические проблемы: построе
ние математических моделей, сбор данных, интерпретаци
и анализ результатов и т. д.;

2) математические проблемы: развитие математически
методов для определенных классов задач;

3) вычислительные проблемы: изучение вычислительны
схем математических методов, совершенствование соответ-
ствующих вычислительных программ и т. д.

Эти три направления, конечно, не являются независи-
мыми. Так, например, математические модели приспосабли-

[1]) Смотри список литературы в конце книги.

ются, насколько это возможно, к существующим методам.
римером может служить линеаризация нелинейных ограни-
ений или оптимизируемой функции с целью применения
етодов линейного программирования. Изучение вычисли-
льных схем математических методов и практика вычисле-
ий часто приводят к улучшению самих методов. Некото-
ые вопросы, такие, как накопление ошибок округления,
ыстрота сходимости, проще всего исследуются эксперимен-
ально. Наконец, опыт вычислений лучше, чем теория, по-
азывает, хорош ли рассматриваемый метод на практике
ли нет. Широкое распространение симплексного метода
сновано не на теоретически доказанной конечности и не на
озможности преодоления вырожденности. Оно основано на
пыте, показывающем, что необходимое число итераций не-
бычайно мало даже для задач больших размеров, что
шибки округления практически менее значительны, чем тео-
ретические оценки, что преодоление вырожденности на
рактике излишне.

Задачи математического программирования могут быть
подразделены на четыре класса:

1) детерминированные непрерывные модели. Совокупность
точек, удовлетворяющих ограничениям (допустимая область),
связна, оптимизируемая функция непрерывна;

2) детерминированные разрывные модели. Допустимая
область несвязна или (и) оптимизируемая функция раз-
рывна;

3) стохастические модели. Коэффициенты ограничений
или (и) оптимизируемой функции — случайные величины;

4) динамические модели. Коэффициенты ограничений или
(и) оптимизируемой функции зависят от параметра (например,
от времени). Задача должна быть решена для каждого зна-
чения параметра.

К первому классу относятся:

а. Линейное программирование. Многие работы в этой
области, как видно из п. 1.1, уже завершены. В настоящее
время, как следует из второй части данной монографии, про-
должают разрабатываться вычислительные алгоритмы сим-
плексного метода. Совсем недавно Данцигом и Вулфом [18]
и независимо от них Бендером [2] был предложен специ-
альный метод решения задач линейного программирования
очень больших размеров, использующий симплексный метод

для решения ряда меньших вспомогательных задач. Эт
метод более эффективен, чем обычный, если ограничен
имеют специальную повторяющуюся структуру.

b. Квадратичное программирование, изучающее зада минимизации выпуклой квадратичной функции при линейн ограничениях. Различными авторами было предложено н сколько методов решения этой задачи. Методы Била [и Вулфа [9] являются прямым расширением симплексного м тода; методы Франка и Вулфа [32] и Зойтендейка [2 (а также гл. 10 настоящей книги) используют симплексны метод для решения ряда вспомогательных задач [1]).

c. Задача минимизации выпуклых функций общего вид при линейных ограничениях. Многие разработанные дл этой задачи методы могут трактоваться как градиентны методы с большим шагом [2]). Упомянем методы Франк и Вулфа [32], Розена [28] и Зойтендейка [20] (а такж гл. 11 настоящей книги). Как было показано Чарнсо и Лемке [38], оптимизируемая функция, представимая в вид суммы выпуклых функций, зависящих каждая от одного аргумента, легко поддается линеаризации. Интересная аналогия между подобными задачами программирования и электрическими цепями установлена Деннисом [19].

d. Выпуклое программирование изучает задачу отыскания минимума выпуклой функции (или максимума вогнутой) на выпуклом множестве. Методы решения этой задачи развивались Эрроу, Гурвицем и Удзавой [40] (градиентный метод с малым шагом [2])), Зойтендейком [20] (а также гл. 11 настоящей книги), Келли, а также Чени и Голдстайном [39]. Предположение выпуклости играет в этих задачах весьма важную роль. Без этого предположения существующие методы либо не пригодны, либо, в лучшем случае, ведут к локальному экстремуму, так что никогда нельзя быть уверенным, что достигнут абсолютный экстремум.

---

[1]) Процесс решения нелинейных задач программирования указанными методами, а также методами возможных направлений, состоит в решении последовательности существенно более простых задач. Отдельную задачу последовательности (subproblem) будем называть вспомогательной. — *Прим. перев.*

[2]) Говоря о большом шаге, автор, по-видимому, имеет в виду конечный шаг. Далее, под малым шагом автор подразумевает бесконечно малый шаг, когда процесс решения задачи градиентным методом описывается дифференциальными уравнениями. — *Прим. ред.*

Ко второму классу относятся, например:

a. Целочисленное линейное программирование. Решение олжно удовлетворять дополнительному условию — состоять з целых чисел. Для случая, когда требование целочисленности относится ко всем переменным, специальные методы ыли разработаны Гомори, Бендером, Кечполем, Квикеном.

b. Смешанные дискретно-непрерывные задачи. Только асть переменных оптимального решения должна удовлетворять условию целочисленности. Многие хорошо известные адачи, такие, как задача коммивояжера, относятся к этому ипу. Методы решения таких задач были предложены Билом Бендером.

К третьему классу относятся задачи со случайными граничениями. Простым примером служит задача линейного программирования со случайным вектором ограничений ли случайным целевым вектором. Хотя некоторые специальные задачи этого типа изучались, в настоящее время не существует общих методов решения широкого класса подобных задач.

Наконец, динамические задачи можно решать, применяя метод динамического программирования Беллмана [1]. Во многих случаях задачу удается сформулировать как статическую, что приводит, однако, к задаче линейного программирования больших размеров.

1.3. Предмет монографии.

В монографии рассматриваются математические и вычислительные аспекты линейного, квадратичного и выпуклого программирования.

Перечислим наиболее важные результаты.

1. Разработан новый вычислительный алгоритм симплексного метода, названный модифицированным мультипликативным алгоритмом (гл. 4). Как показано в гл. 5, при этом алгоритме на одну итерацию идет меньше вычислительного времени, чем при любом существующем алгоритме.

2. Разработано несколько методов выпуклого программирования, названных методами возможных направлений (гл. 7). Эти методы конечны в линейном и квадратичном случаях (гл. 9 и 10).

… состоящей из … …учаются математические ос… …ого программирования, книга состо… …ся. Во второй части (гл. 3—6) рассматрива… задачи линейного программирования. После краткого … сания симплексного метода, двойственного симплексного ме… тода и метода одновременного решения прямой и двойствен… ной задач (гл. 3) рассматриваются вычислительные схемы первых двух методов (гл. 4). Кроме существующих алгоритмов — прямого, алгоритма с обратной матрицей и мультипликативного, — описывается новый модифицированный мультипликативный алгоритм. В этом алгоритме лучше используется тот факт, что лишь небольшое число элементов матриц практических задач линейного программирования отлично от нуля. Более эффективно в этом алгоритме выполняются повторения и вычисления относительных оценок. Еще одно усовершенствование состоит в сокращении счета особым рассмотрением векторов, то входящих в базис, то покидающих его. Теоретическое сравнение различных алгоритмов в гл. 5 показывает, что новый алгоритм превосходит остальные в быстроте выполнения итераций. Число итераций при использовании нового алгоритма не сокращается. Разработка приемов, сокращающих число итераций, может быть важным предметом будущих исследований. Наконец, гл. 6 связывает две части книги, поскольку она посвящена некоторым специальным задачам линейного программирования, возникающим в качестве вспомогательных задач в нелинейном программировании (третья часть). В гл. 6 указывается, как применять модифицированный мультипликативный алгоритм для решения таких задач.

В третьей части книги изучаются методы возможных направлений.

Это итерационные методы решения задачи максимизации вогнутой функции на выпуклом множестве, удовлетворяющие следующим правилам:

1) исходное решение, с которого начинается последовательность итераций, должно быть возможным, т. е. удовлетворять всем ограничениям;

… …ного решения к новому возможному решению с большим значением оптимизируемой функции. Этот процесс продолжается, пока возможно дальнейшее увеличение значения оптимизируемой функции;

3) последовательность значений оптимизируемой функции сходится к условному максимуму функции, если этот максимум существует. В противном случае последовательность не ограничена.

Второе правило может быть подразделено на два:

2a) определяется подходящее возможное направление, т. е. направление, в котором можно сделать шаг конечной длины, оставаясь в пределах допустимой области, и достигнуть большего значения оптимизируемой функции;

2b) определяется длина шага в этом направлении.

Методы возможных направлений могут рассматриваться как градиентные методы с большим шагом[1]). Не следует смешивать их с градиентными методами с малым шагом[1]), которые также успешно используются для решения задач выпуклого программирования [40]. Ясно, что многие существующие методы в действительности являются методами возможных направлений.

Глава 7 посвящена общей теории методов возможных направлений. В ней показано, что различные нормализации при выборе возможного направления приводят к различным методам решения. Таким образом, возникает интересная аналогия с результатами Хестенса и Штифеля [34], показавших, что некоторые существующие методы решения систем линейных уравнений являются реализацией метода сопряженных градиентов и различаются лишь способами фиксации направления с помощью дополнительных требований. В гл. 7 описываются пять различных методов нормализации. Нетрудно указать и другие методы. При каждой нормализации мы получаем задачу выбора возможного направления, разрешимую симплексным или двойственным симплексным методом. Кроме того, в гл. 7 рассматривается определение исходного возможного решения, определение длины шага, доказательство сходимости и программирование без предположения о выпуклости.

---

[1]) См. примечание 2 на стр. 10. — *Прим. ред.*

Вычислительные схемы, возникающие при различных нормализациях, изучаются в гл. 8. Там показывается, что для решения задачи выбора направления может успешно применяться модифицированный мультипликативный алгоритм. Показано также, как использовать результаты предыдущего выбора направления для ускорения решения задачи следующего выбора.

Главы 9, 10 и 11 посвящены приложению методов возможных направлений к линейному, квадратичному и выпуклому программированию соответственно. В каждой из них кратко описываются алгоритмы решения и получены следующие результаты:

1) для квадратичного программирования получено несколько конечных методов, использующих принцип сопряженных направлений;

2) одна из нормализаций ведет к методу, эквивалентному симплексному в случае линейного программирования, и методу Била в случае квадратичного. Метод решения, основанный на этой нормализации, можно поэтому рассматривать как обобщение симплексного метода на нелинейные задачи программирования;

3) другая нормализация ведет к методу, эквивалентному методу одновременного решения прямой и двойственной задач. Следовательно, соответствующий алгоритм можно рассматривать как обобщение метода на нелинейные задачи программирования;

4) методы возможных направлений в предлагаемой здесь форме можно использовать для решения задач линейного программирования очень больших размеров путем решения последовательности меньших вспомогательных задач. Эти методы способствуют разрешению важной проблемы — проблемы решения линейных задач больших размеров, имеющих специальную структуру.

1.4. Обозначения.

Повсюду в книге приняты матричные обозначения. Матрицы обозначаются прописными буквами, векторы-столбцы — латинскими строчными, для обозначения векторов-строк используется знак транспонирования T. Исключение составляет описание вычислительных схем симплексного метода, где в согласии с обозначениями, принятыми в литературе по данному вопросу, некоторые векторы обозначены греческими буквами. Во всех других случаях греческими буквами обозначаются скаляры. Векторное неравенство $x \leqslant y$ означает, что каждая компонента вектора x не превосходит соответствующей компоненты вектора y. В неравенстве $x \geqslant 0$ символ 0 означает вектор, все компоненты которого равны нулю. Компоненты вектора x обозначаются x_j, а x^ν означает ν-й вектор последовательности (с компонентами x_j^ν). Множество может быть обозначено перечислением всех его элементов в фигурных скобках, например $I = \{1, \ldots, m\}$, или заданием условий, его определяющих, например $R = \{x \mid Ax \leqslant b,\ x \geqslant 0\}$.

Если $F(x)$ — функция вектора x и если R — множество векторов (точек) x, тогда $\max\{F(x) \mid x \in R\}$ обозначает верхнюю грань функции $F(x)$ при условии, что $x \in R$. Строго говоря, термин „max“, можно использовать лишь тогда, когда известно, что существует вектор $x \in R$, на котором $F(x)$ достигает верхней грани. Однако мы будем употреблять этот термин и тогда, когда существование такого x не известно, как принято в литературе по рассматриваемому нами вопросу. Мы будем говорить о задаче математического программирования $\max\{F(x) \mid x \in R\}$, имея в виду следующие вопросы:

1) существует ли вектор x, принадлежащий R;

2) если R непусто, ограничена ли на R функция $F(x)$;

3) если $F(x)$ ограничена на R, каково значение максимума и на каком x этот максимум достигается. Если такой вектор x не единствен, нам достаточно определить хотя бы один. В некоторых случаях используются операторы $\forall$ и $\exists$ [1]:

„$\forall_x (x \in R)\, F(x) \leqslant m$“ означает, что для всех x, принадлежащих R, имеет место неравенство $F(x) \leqslant m$;

„$\exists_x\ x \in R,\ F(x) \leqslant m$“ означает, что существует вектор x, принадлежащий R, для которого $F(x) \leqslant m$.

---

[1]) $\forall$ — перевернутая первая буква английского слова All, $\exists$ — перевернутая первая буква английского слова Existense. — *Прим. перев.*

Глава 2

ТЕОРИЯ ВЫПУКЛОГО ПРОГРАММИРОВАНИЯ

2.1. Введение.

Как отмечалось в гл. 1, задача выпуклого про граммирования состоит в отыскании минимума выпукло функции, заданной на выпуклом множестве (или максимум вогнутой функции, также заданной на выпуклом множестве. В этой главе изучаются математические основы такой задачи. В первых трех пунктах кратко рассматривается теория ли нейных неравенств, полуопределенных матриц и выпуклых функций. Здесь не делается попытки полного обзора эти областей. Пункты 2.5 и 2.6 посвящены соответственно зада чам линейного и выпуклого программирования.

2.2. Теория линейных неравенств.

Пусть A — прямоугольная матрица $m \times n$, p и x — n-мерные векторы; элементы A, x и p являются действительными числами, так же как и элементы всех матриц и векторов, вводимых в дальнейшем. Строки A, векторы p и x можно рассматривать как точки в n-мерном эвклидовом пространстве E_n. Как указывалось выше, значком T мы будем обозначать транспонирование.

Основной теоремой теории линейных неравенств является теорема Фаркаша [30]:

Теорема 1. *Если $p^Tx \geqslant 0$ ($p^Tx \leqslant 0$) для всех x, удовлетворяющих системе линейных неравенств $Ax \geqslant 0$ ($Ax \leqslant 0$), то p можно представить в виде линейной комбинации с неотрицательными коэффициентами строк матрицы A, $p = A^Tu$, где $u \geqslant 0$ — m-мерный вектор.*

Доказательство. Пусть $C = \{y \in E_n | \exists_u u \geqslant 0,\ y = A^Tu\}$. Тогда множество C — выпуклый многогранный конус

в n-мерном пространстве. Поэтому если $y \in C$, то $\lambda y \in C$ для всех $\lambda \geqslant 0$, если $y_1 \in C$ и $y_2 \in C$, то $y_1 + y_2 \in C$. Покажем, что если вектор p не принадлежит C, то существует вектор x, для которого $Ax \geqslant 0$, но $p^Tx < 0$. Этим теорема будет доказана. Итак, предположим, что $p \notin C$. Пусть q — проекция [1] p на C и $r = q - p$. Тогда:

1) для всех $y \in C$ $(q - p)^T (y - q) \geqslant 0$, поскольку в противном случае на отрезке, соединяющем y и q, найдется точка, лежащая ближе к p, чем q. Так как эта точка принадлежит C, то q не может быть проекцией p на C;

2) $r^Tq = 0$, так как в противном случае на прямой, соединяющей начало координат и q, найдется точка, лежащая ближе к p, чем q. Поскольку эта точка принадлежит C, то q не может быть проекцией p на C.

Из 1 и 2 следует, что $r^Ty \geqslant 0$ для всех $y \in C$. Если $a_{i\cdot}^T$ — i-я строка матрицы A, то $a_{i\cdot}^T = e_i^T A$ (e_i — i-й единичный вектор). Следовательно, точка $a_{i\cdot} \in C$, т. е. $r^T a_{i\cdot} \geqslant 0$ или $a_{i\cdot}^T r \geqslant 0$. Здесь i произвольно, поэтому $Ar \geqslant 0$. Поскольку $p^Tr = q^Tr - r^Tr = - r^Tr < 0$, то r — искомый вектор.

Теорема 2. *Система линейных неравенств $Ax \geqslant 0$, $A^Tu = 0$, $u \geqslant 0$, всегда имеет решение x', u', удовлетворяющее условиям $Ax' + u' > 0$ и $u_i' a_{i\cdot}^T x' = 0$ для всех i.*

Доказательство. Если $a_{i\cdot}^T x \leqslant 0$ для всех x, удовлетворяющих неравенствам $Ax \geqslant 0$, то, по теореме Фаркаша, найдется вектор $u \geqslant 0$, такой, что $-a_{i\cdot}^T = A^Tu$, т. е. найдется такой вектор $u^i \geqslant 0$, что $A^Tu^i = 0$ и $u_i^i > 0$. Отсюда следует, что либо $a_{i\cdot}^T x^i > 0$ для некоторого x^i, удовлетворяющего условиям $Ax^i \geqslant 0$, либо $A^Tu^i = 0$ для некоторого вектора $u^i \geqslant 0$ с компонентой $u_i^i > 0$. В первом случае положим $u^i = 0$, во втором $x^i = 0$. Тогда для всех i

---

[1]) q — точка множества C, наименее удаленная от p. Для ее существования необходима замкнутость множества C. Можно показать, что множество C замкнуто. Исходя из приведенного определения выпуклого многогранного множества C, доказательство его замкнутости нетривиально. — *Прим. ред.*

ЖИТОМИРСЬКА ОБЛ. БІБЛІОТЕКА

выполняется условие $a_i^T.x^i+u_i^i>0$. Положим далее $x'=\sum_{i=1}^{m}$ и $u'=\sum_{i=1}^{m} u^i$, тогда:

1) $Ax' \geqslant 0$, поскольку $Ax^i \geqslant 0$ для всех i;
2) $A^Tu'=0$, поскольку $A^Tu^i=0$ для всех i;
3) $u' \geqslant 0$, поскольку $u^i \geqslant 0$ для всех i;
4) $Ax'+u'>0$, поскольку $a_i^T.x^r+u_i^r \geqslant 0$ для всех r, а при $r=i$ выполняется строгое неравенство;
5. $u_i'a_i^T.x'=0$, поскольку $u_i^h a_i^T.x'=0$ для всех h. Теорема доказана.

Эта теорема была впервые сформулирована Таккером [29] и названа им ключевой. Прямое элементарное доказательство теоремы приводится у Гуда [12]. Опираясь на теорему 2 нетрудно доказать теорему Фаркаша и многие другие теоремы линейных неравенств.

Примером служит следующее утверждение.

Теорема 3. *Система линейных неравенств $Ax<0$ несовместна тогда и только тогда, когда существует решение системы уравнений $A^Tu=0$, удовлетворяющее условиям $u \geqslant 0$ и $u_i>0$ по крайней мере для одного i, т. е. когда одна из строк матрицы A представляется в виде линейной комбинации других с неположительными коэффициентами.*

Доказательство. В силу теоремы 2 существуют векторы x' и u', удовлетворяющие условиям $-Ax' \geqslant 0$, $A^Tu'=0$, $u' \geqslant 0$ и $-a_i^T.x'+u_i'>0$. Если для всех i выполняется условие $u_i'=0$, то система $a_i^T.x<0$ разрешима. Решением является, например, вектор x'. Отсюда ясно, что если эта система неразрешима, то в любом решении системы $A^Tu=0$, $u \geqslant 0$, одна из компонент u строго положительна. Этим доказана необходимость условий теоремы. С другой стороны, если вектор u удовлетворяет условиям $A^Tu=0$, $u \geqslant 0$, $u_i>0$ для некоторого i, то $x^TA^Tu=0$, или $u^TAx=0$ для всех x. Поэтому $a_i^T.x=0$ для x, удовлетворяющих системе $Ax \leqslant 0$, если $u_i>0$. Следовательно, не существует такого x, что $a_i^T.x<0$ для всех i, т. е. рассматриваемая система несовместна.

Следствие. Из теоремы 2 следует возможность такого выбора вектора u [1]), что $u_i>0$ для тех i, для которых равенство $a_i^T.x=0$ выполняется для всех x, удовлетворяющих системе $Ax \leqslant 0$.

2.3. Определенные матрицы.

Квадратная симметрическая матрица C называется положительно определенной, если $s^TCs>0$ для любого вектора $s \neq 0$; неотрицательно определенной, если $s^TCs \geqslant 0$ для любого вектора s; отрицательно (неположительно) определенной, если матрица $-C$ положительно (неотрицательно) определена.

Теорема 1. *Произвольный главный минор положительно (неотрицательно) определенной матрицы также положительно (неотрицательно) определен.*

Доказательство. Пусть главный минор P матрицы C состоит из строк и столбцов, номера которых $i \in I_1 \subset I = \{1, \ldots, m\}$. Пусть s_i произвольно, если $i \in I_1$, и равно нулю, если $i \notin I_1$. Тогда

$$0 \leqslant s^TCs=\sum_{i=1}^{m}\sum_{j=1}^{m} c_{ij}s_is_j=\sum_{i\in I_1}\sum_{j\in I_1} p_{ij}s_is_j.$$

Если $s \neq 0$ и C положительно определена, то выполняется строгое неравенство. Теорема доказана.

Следствие. Диагональные элементы положительно (неотрицательно) определенной матрицы положительны (неотрицательны).

Теорема 2. *Если C неотрицательно определена и $s^TCs=0$, то $Cs=0$.*

Доказательство. Если $s^TCs=0$, то для всех m-мерных векторов t и скаляров λ выполняется неравенство

$$0 \leqslant (t+\lambda s)^TC(t+\lambda s)=t^TCt+2\lambda t^TCs,$$

т. е. $t^TCs=0$ для любого t. Следовательно, $Cs=0$.

[1]) Предполагается, что $A^Tu=0$. — *Прим. ред.*

Теорема 3. *Если C — положительно определенная матрица, то обратная матрица C^{-1} существует и положительно определена.*

Доказательство. Пусть C — положительно определенная матрица. Предположим, что определитель C равен нулю, так что система линейных уравнений $Cs = 0$ имеет нетривиальное решение s'. Тогда $(s')^T Cs' = 0$, $s' \neq 0$, что противоречит предположению о положительной определенности C. Таким образом, C^{-1} существует. Возьмем теперь произвольный вектор $s \neq 0$ и положим $t = C^{-1}s$. Тогда $t \neq 0$ и $s^T C^{-1} s = s^T C^{-1} C C^{-1} s = t^T C t > 0$, т. е. C^{-1} — положительно определенная матрица.

Теорема 4. *Если A — произвольная (быть может прямоугольная) матрица, то матрицы AA^T и $A^T A$ неотрицательно определены.*

Доказательство. Положим $t = A^T s$. Тогда

$$s^T A A^T s = (s^T A)(A^T s) = t^T t \geqslant 0.$$

Теорема 5. *Симметричная идемпотентная матрица неотрицательно определена.*

Доказательство. Пусть A симметрична и идемпотентна, т. е. $A^T = A$ и $A^2 = A$, тогда $A = A^T A$. Отсюда, в силу теоремы 4, следует неотрицательная определенность.

Теорема 6. *Пусть неотрицательно определенная матрица C представима в виде $C = \begin{pmatrix} P & Q \\ Q^T & S \end{pmatrix}$, где P — положительно определенная матрица порядка n_1, S — неотрицательно определенная матрица порядка n_2 $(n_1 + n_2 = n)$, Q — матрица $n_1 \times n_2$. Тогда матрица $-Q^T P^{-1} Q + S$ порядка n_2 неотрицательно определена.*

Доказательство. Пусть $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, где x_1 — n_1-мерный вектор, а x_2 — n_2-мерный вектор. В силу неотрицательной определенности матрицы C,

$$0 \leqslant (x_1^T,\ x_2^T)\begin{pmatrix} P & Q \\ Q^T & S \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = x_1^T P x_1 + 2x_1^T Q x_2 + x_2^T S x_2.$$

Положим $x_1 = -P^{-1} Q x_2$. Тогда для произвольного вектора x_2

$$x_2^T Q^T P^{-1} Q x_2 - 2x_2^T Q^T P^{-1} Q x_2 + x_2^T S x_2 \geqslant 0,$$

или $x_2^T(-Q^T P^{-1} Q + S)\, x_2 \geqslant 0$, что доказывает теорему.

Пусть C — положительно определенная матрица, p — n-мерный вектор. Рассмотрим систему линейных уравнений $Cx = p$ и определим следующую последовательность операций:

1) возьмем произвольный вектор x^0, обозначим $g^0 = p - Cx^0$ и, выбрав вектор $s^0 \neq 0$, положим

$$\lambda_0 = \frac{(g^0)^T s^0}{(s^0)^T C s^0} \quad \text{и} \quad x^1 = x^0 + \lambda_0 s^0; \tag{2.3.1}$$

2) определив точки x^h, векторы s^h и скаляры λ_h для $h = 0, 1, \ldots, k-1$, найдем

$$x^k = x^{k-1} + \lambda_{k-1} s^{k-1} \tag{2.3.2}$$

и

$$g^k = p - Cx^k. \tag{2.3.3}$$

В качестве s^k выбираем вектор, удовлетворяющий условиям

$$(s^h)^T C s^k = 0, \quad h = 0, 1, \ldots, k-1, \quad s^k \neq 0, \tag{2.3.4}$$

если такой вектор существует, и полагаем

$$\lambda_k = \frac{(g^k)^T s^k}{(s^k)^T C s^k}. \tag{2.3.5}$$

Если такого вектора не существует, построение последовательности заканчивается.

Теорема 7. *Векторы s^h, порожденные указанной последовательностью операций, линейно независимы.*

Доказательство. Предположим, что $\sum_0^p u_h s^h = 0$ и $\sum_0^{p-1} u_h s^h \neq 0$, так что $u_p \neq 0$. Тогда $\sum_0^p u_h C s^h = 0$. Поскольку

$(s^p)^T Cs^h = 0$ для $h = 0, 1, \ldots, p-1$, то $u_p (s^p)^T Cs^p =$
Учитывая, что $u_p \neq 0$, получаем $(s^p)^T Cs^p = 0$, т. е. $s^p =$
Теорема доказана.

Следствие. Построение определенной выше последовательности заканчивается за конечное число k ($k \leqslant n$) шагов.

Теорема 8. *Пусть* $\bar{x} = x^0 + \sum_{h=0}^{k-1} \lambda_h s^h$, *т. е.* $\bar{x} = x$

Тогда $\bar{x}$ — *решение системы линейных уравнений* $Cx =$

Доказательство. Пусть $\bar{g} = p - C\bar{x}$. Предположим, что $\bar{g} \neq 0$, т. е. существует вектор s, такой, что $\bar{g}^T s \neq$ (например, $s = \bar{g}$). Положим $s = \sum_{h=0}^{k-1} \mu_h s^h + t$, где

$$\mu_h = \frac{(s^h)^T Cs}{(s^h)^T Cs^h}.$$

Тогда

$$(s^l)^T Ct = (s^l)^T Cs - \sum_{h=0}^{k-1} \mu_h (s^l)^T Cs^h = (s^l)^T Cs - \mu_l (s^l)^T Cs^l = 0$$

$$l = 0, 1, \ldots, k-1.$$

Заметим теперь, что для всех l $\bar{g} = g^l - \sum_{h=l}^{k-1} \lambda_h Cs^h$, следовательно,

$$\bar{g}^T s^l = (g^l)^T s^l - \lambda_l (s^l)^T Cs^l = 0 \quad \text{для} \quad l = 0, 1, \ldots, k-1.$$

Поэтому $\bar{g}^T s = \bar{g}^T t \neq 0$, и вектор t удовлетворяет условиям $t \neq 0$ и $(s^l)^T Ct = 0$ для $l = 0, 1, \ldots, k-1$. Это противоречит предположению о том, что для x^k такого вектора не существует. Таким образом, неравенство $\bar{g} = g^k \neq 0$ не может иметь места, что доказывает теорему.

Направления s^h называются сопряженными. Они определяются неединственным образом. Многие методы решения систем линейных уравнений по существу используют сопряженные направления и различаются лишь способами фиксации векторов s^h с помощью дополнительных требований. Это было показано Хестенсом и Штифелем [34]. К таким методам принадлежит метод исключения Гаусса и метод сопряженных градиентов [1]).

2.4. Выпуклые функции и области.

Пусть $F(x) = F(x_1, \ldots, x_n)$ — функция n действительных переменных $x_1, \ldots, x_n$, т. е. вектора $x \in E_n$. Будем предполагать, что $F(x)$ дифференцируема и частные производные ее непрерывны. Из этого следует, что если градиент функции в точке x обозначить через $g(x)$, т. е.

$$\{g(x)\}^T = \left\{\frac{\partial F}{\partial x_1}, \ldots, \frac{\partial F}{\partial x_n}\right\}, \tag{2.4.1}$$

компоненты вектора $g(x)$ — непрерывные функции x.

Будем называть функцию $F(x)$ выпуклой, если для произвольных векторов x и y и скаляра λ, $0 < \lambda < 1$, выполняется неравенство

$$F((1-\lambda)x + \lambda y) \leqslant (1-\lambda)F(x) + \lambda F(y). \tag{2.4.2}$$

Будем называть функцию $F(x)$ вогнутой, если $-F(x)$ выпукла. Вогнутая функция удовлетворяет неравенству

$$F((1-\lambda)x + \lambda y) \geqslant (1-\lambda)F(x) + \lambda F(y). \tag{2.4.3}$$

Функция $F(x)$ строго выпукла (вогнута), если при $x \neq y$ в (2.4.2), (2.4.3) выполняется строгое неравенство. Ясно, что линейная функция является и выпуклой и вогнутой, но не строго.

Справедливы следующие теоремы:

Теорема 1. *$F(x)$ выпукла тогда и только тогда когда функция одной переменной $\varphi(\lambda) = F(x + \lambda s)$ выпукла для произвольных векторов x и s,*

Доказательство. Необходимость условий теоремы очевидна. Предположим теперь, что $\varphi(\lambda)$ выпукла для любых двух векторов x, s. Выберем произвольно x, $y \in E_n$ и

[1]) См., например, Березин И. С. и Жидков Н. П., Методы вычислений, т. II, Физматгиз, М., 1959. — *Прим. перев.*

положим $y - x = s$, тогда

$$F((1-\lambda)x + \lambda y) = F(x + \lambda s) = \varphi(\lambda) =$$
$$= \varphi((1-\lambda).0 + \lambda.1) \leqslant (1-\lambda)\varphi(0) + \lambda\varphi(1) =$$
$$= (1-\lambda)F(x) + \lambda F$$

Теорема доказана.

Теорема 2. *Если* $x^h \in E_n$, $h = 1, \ldots, m$, *и* F *выпукла, то*

$$F\left(\frac{1}{m}\sum_{h=1}^{m} x^h\right) \leqslant \frac{1}{m}\sum_{h=1}^{m} F(x^h).$$

Доказательство. Теорема справедлива для $m =$ и $m = 2$. Предположим, что она верна для некоторого и докажем, что тогда она верна для $m+1$:

$$F\left(\frac{1}{m+1}\sum_{h=1}^{m+1} x^h\right) = F\left(\frac{m}{m+1}\,\frac{1}{m}\sum_{h=1}^{m} x^h + \frac{1}{m+1}\,x^{m+1}\right) \leqslant$$
$$\leqslant \frac{m}{m+1}F\left(\frac{1}{m}\sum_{h=1}^{m} x^h\right) + \frac{1}{m+1}F(x^{m+1}) \leqslant \frac{1}{m+1}\sum_{h=1}^{m+1} F(x^h)$$

Теорема доказана.

Теорема 3. *Следующие утверждения равносильны*
1. $F(x)$ *выпукла;*
2. $F(x_2) - F(x_1) \geqslant g(x_1)^T(x_2 - x_1)$ *для любых двух векторов* $x_1, x_2 \in E_n$;
3. $g(x + \lambda s)^T s$ — *неубывающая функция* λ *при произвольных* x, s.

Пусть существуют все частные производные второго порядка функции $F(x)$, *образующие матрицу* $C(x)$. *Тогда*
4. $C(x)$ *неотрицательно определена для любого* $x \in E_n$.

Доказательство $1 \to 2$. Пусть $F(x)$ выпукла. Тогда для любых двух векторов x_1, $x_2 \in E_n$

$$F((1-\lambda)x_1 + \lambda x_2) \leqslant (1-\lambda)F(x_1) + \lambda F(x_2), \qquad 0 < \lambda < 1,$$

или

$$F(x_1 + \lambda(x_2 - x_1)) \leqslant F(x_1) + \lambda(F(x_2) - F(x_1)),$$

т. е.

$$\frac{F(x_1 + \lambda(x_2 - x_1)) - F(x_1)}{\lambda} \leqslant F(x_2) - F(x_1). \quad (2.4.4)$$

Переходя к пределу при $\lambda \to 0$, получаем

$$g(x_1)^T(x_2 - x_1) \leqslant F(x_2) - F(x_1). \quad (2.4.5)$$

$2 \to 3$. $\varphi(\lambda) = F(x + \lambda s)$ выпукла, так что из (2.4.5) следует

$$\varphi'(\lambda_1)(\lambda_2 - \lambda_1) \leqslant \varphi(\lambda_2) - \varphi(\lambda_1),$$
$$\varphi'(\lambda_2)(\lambda_1 - \lambda_2) \leqslant \varphi(\lambda_1) - \varphi(\lambda_2).$$

Тогда при $\lambda_2 > \lambda_1$ получим

$$\varphi'(\lambda_1) \leqslant \frac{\varphi(\lambda_2) - \varphi(\lambda_1)}{\lambda_2 - \lambda_1} \leqslant \varphi'(\lambda_2), \quad (2.4.6)$$

т. е. $\varphi'(\lambda_1) \leqslant \varphi'(\lambda_2)$. Таким образом, $g(x + \lambda_1 s)^T s \leqslant g(x + \lambda_2 s)^T s$ при $\lambda_1 < \lambda_2$, что и требовалось доказать.

$3 \to 1$. Предположим, что $g(x + \lambda s)^T s$ — неубывающая функция λ для всех x и s. Тогда $\varphi'(\lambda_1) \leqslant \varphi'(\lambda_2)$ при $\lambda_1 < \lambda_2$.

Если $0 < \mu < 1$ и $\lambda_2 > \lambda_1$, то

$$0 \leqslant \mu(\lambda_2 - \lambda_1)\int_0^1 d\tau\,[\varphi'(\lambda_1 + \tau(\lambda_2 - \lambda_1)) - \varphi'(\lambda_1 + \mu\tau(\lambda_2 - \lambda_1))] =$$
$$= (1-\mu)\varphi(\lambda_1) + \mu\varphi(\lambda_2) - \varphi((1-\mu)\lambda_1 + \mu\lambda_2).$$

Отсюда следует выпуклость $\varphi(\lambda)$, а значит, и $F(x)$.

$3 \to 4$. Из (2.4.6) получим $\varphi''(\lambda) \geqslant 0$ для всех λ, т. е. $s^T C(x + \lambda s)s \geqslant 0$ для всех x, s и λ. Следовательно, $s^T C(x)s \geqslant 0$ для всех x и s, значит, $C(x)$ неотрицательно определена для всех x.

$4 \to 3$. Если $C(x)$ неотрицательно определена, то $s^T C(x + \lambda s)s \geqslant 0$ для всех x, s и λ, т. е. $\varphi''(\lambda) \geqslant 0$ для всех λ. Отсюда непосредственно следует, что $\varphi'(\lambda)$ — неубывающая функция.

Следствия. 1. Если $F(x_2) < F(x_1)$ и $F(x)$ — выпуклая функция, то $g(x_1)^T(x_2 - x_1) < 0$ [следует из (2.4.5)].

2. Если $g(x_1)^T(x-x_1)\geqslant 0$ для всех x и $F(x)$ — выпуклая функция, то $F(x)$ достигает минимума в точке

3. Квадратичная функция $F(x)=p^Tx+\frac{1}{2}x^TCx$ выпук... тогда и только тогда, когда симметрическая матрица ... порядка n неотрицательно определена.

Теорема 4. *Линейная комбинация с положите... ными коэффициентами выпуклых функций выпук... и строго выпукла, если по крайней мере одна из исх... ных функций строго выпукла.*

Доказательство непосредственно следует из определен... выпуклости.

Пусть I — конечное или бесконечное множество индексо... a_i — вектор, b_i — скаляр для любого $i\in I$.

Теорема 5. *Если* $F(x)=\sup\{a_i^Tx-b_i\mid i\in I\}$, *т... $F(x)$ — выпуклая функция.*

Доказательство. Возьмем $x_1,\ x_2\in E_n$ и $0<\lambda<$... Тогда

$$F((1-\lambda)x_1+\lambda x_2)=\sup_i\{a_i^T((1-\lambda)x_1+\lambda x_2)-b_i\}\leqslant$$
$$\leqslant(1-\lambda)\sup_i\{a_i^Tx_1-b_i\}+\lambda\sup_i\{a_i^Tx_2-b_i\}=$$
$$=(1-\lambda)F(x_1)+\lambda F(x_2)$$

Определение. *Множество R в n-мерном эвклидо... вом пространстве называется выпуклым, если вмест... с двумя произвольными точками $x_1\in R$ и $x_2\in R$ он... содержит соединяющий их отрезок, т. е. R — выпукло... множество, если*

$$\forall_{x_1,x_2}(x_1,\ x_2\in R)\{x\mid\exists_\lambda,\ 0\leqslant\lambda\leqslant 1,\ x=(1-\lambda)x_1+\lambda x_2\}\subset R$$

Имеют место следующие теоремы:

Теорема 6. *Пересечение любого числа выпуклых множеств выпукло.*

Доказательство. Пусть R_i — выпуклое множеств... при $i\in I$, $x_1,\ x_2\in\bigcap R_i$, $0\leqslant\lambda\leqslant 1$. Тогда x_1 и x_2, а значит

$(1-\lambda)x_1+\lambda x_2$ принадлежат множеству R_i для всех i. Следовательно, $(1-\lambda)x_1+\lambda x_2\in R_i$. Теорема доказана.

Теорема 7. *Если $f_i(x)$ — выпуклая функция $x\in E_n$ для $i\in I$, а b_i — скаляр, то $R=\{x\mid\forall_i(i\in I)\,f_i(x)\leqslant b_i\}$ — замкнутое выпуклое множество.*

Доказательство. Положим $R_i=\{x\mid f_i(x)\leqslant b_i\}$, тогда $R=\bigcap R_i$, и достаточно доказать, что множества R_i замкнуты и выпуклы. Пусть x_1 и x_2 — произвольные точки R_i, $0\leqslant\lambda\leqslant 1$. Тогда $f_i((1-\lambda)x_1+\lambda x_2)\leqslant(1-\lambda)f_i(x_1)+\lambda f_i(x_2)\leqslant b_i$, т. е. $(1-\lambda)x_1+\lambda x_2\in R_i$. Если $x_j\in R_i$, $j=1,\ 2,\ \ldots$ и x — предельная точка этой последовательности, то из $f_i(x_j)\leqslant b_i$ следует $f_i(x)\leqslant b_i$. Поэтому $x\in R_i$ и R_i — замкнутое множество.

Теорема 8. *Если R и $F(x)$ выпуклы, то $\{x\in R\mid F(x)\leqslant m\}$ — выпуклое (быть может, пустое) подмножество R для каждого числа m.*

Доказательство. $R_1=\{x\mid F(x)\leqslant m\}$ — выпуклое множество для любого m. Далее, $\{x\in R\mid F(x)\leqslant m\}=R\cap R_1$.

Теорема 9. *Всякий локальный минимум выпуклой функции на выпуклом множестве является абсолютным минимумом. Множество этих минимумов выпукло.*

Доказательство. Предположим, что $x_1\in R$ и $x_2\in R$ — локальные минимумы и $F(x_2)<F(x_1)$. Тогда из следствия 1 теоремы 3 вытекает, что $g(x_1)^T(x_2-x_1)<0$, т. е. $F(x)$ убывает при движении из x_1 к x_2[1]. Поскольку отрезок, соединяющий x_1 и x_2, принадлежит R, x_1 не может быть локальным минимумом. Следовательно, $F(x_2)=F(x_1)$. Второе утверждение непосредственно следует из теоремы 8.

[1]) Этот же вывод может быть сделан и без предположения о дифференцируемости $F(x)$:

$F(x_1+\lambda(x_2-x_1))\leqslant\lambda F(x_2)+(1-\lambda)F(x_1)<F(x_1)$ $(0<\lambda\leqslant 1)$. — *Прим. ред.*

Теорема 10. *Строго выпуклая функция либо дост гает минимума только в одной точке выпуклого м жества, либо неограничена снизу.*

Доказательство. Пусть $F(x)$ — строго выпук функция и $F(x_1)=F(x_2)$ для $x_1 \in R$ и $x_2 \in R$. Из пред ложения строгой выпуклости следует, что в любой то отрезка, соединяющего x_1 и x_2, значение функции мен ее значения в крайних точках. Таким образом, не существ двух точек, в которых $F(x)$ достигает минимума.

Теорема 11. *Пусть $f_i(x)$ — выпуклая функци $i=1, \ldots, m$, градиент $q_i(x)$ которой — непрерывн вектор-функция, $R=\{x \mid f_i(x) \leqslant b_i, \quad i=1, \ldots, m\}$ пусто. Система неравенств $f_i(x) < b_i$ несовместна тог и только тогда, когда существуют такие неотриц тельные числа u_i, $\sum u_i = 1$, что $\sum u_i q_i(x) = 0$, $\sum u_i f_i(x)$ $=\sum u_i b_i$ для всех $x \in R$.*

Доказательство. Если существуют такие неотриц тельные числа u_i, $\sum u_i = 1$, что $\sum u_i f_i(x) = \sum u_i b_i$ дл всех $x \in R$, то система $f_i(x) < b_i$, $i=1, \ldots, m$, нес вместна. Действительно, если для некоторого $\bar{x}$ неравенст $f_i(\bar{x}) < b_i$ выполняется для всех i и $u_r > 0$, то $u_r f_r(\bar{x}) < u_r b$ $u_i f_i(\bar{x}) \leqslant u_i b_i$ для $i \neq r$, т. е. $\sum u_i f_i(\bar{x}) < \sum u_i b_i$.

Предположим теперь, что система $f_i(x) < b_i$, $i=1, \ldots, m$ несовместна. Возьмем точку $y \in R$ и положим $f_i(y) = b_i$ дл $i \in I_1$; $f_i(y) < b_i$ для $i \in I_2$, $I_1 + I_2 = \{1, \ldots, m\}$. Предпол жим далее, что система $q_i(y)^T s < 0$, $i \in I_1$, совместна. Ясн что в этом случае для некоторого $\lambda > 0$ и всех i выпол няются неравенства $f_i(y+\lambda s) < f_i(y) \leqslant b_i$, что противоречи нашему предположению. Следовательно, система линейны неравенств $q_i(y)^T s < 0$, $i \in I_1$, несовместна. В силу теоремы п. 2.2, можно найти такие неотрицательные числа u_i, сред которых по крайней мере одно $u_i > 0$ или, что эквивалентн $\sum_1^m u_i = 1$, что $\sum_{i \in I_1} u_i q_i(y) = 0$ или $\sum_{i=1}^m u_i q_i(y) = 0$, $u_i = 0$ когда $f_i(y) < b_i$. Поэтому $\sum_{i=1}^m u_i f_i(y) = \sum_{i=1}^m u_i b_i$. Из теоремы 3

ункция $\sum u_i f_i(x)$ выпукла, ее градиент равен нулю точке $y \in R$, так что в этой точке достигается минимум ледствие 2 теоремы 3). Отсюда для всех x выполняется равенство $\sum u_i f_i(x) \geqslant \sum u_i f_i(y) = \sum u_i b_i$. С другой сто оны, $\sum u_i f_i(x) \leqslant \sum u_i b_i$ для $x \in R$, т. е. $\sum u_i f_i(x) = \sum u_i b_i$ ля всех $x \in R$, $\sum u_i = 1$. Более того, поскольку в произ ольной точке $x \in R$ достигается безусловный минимум функ ии $\sum u_i f_i(x)$, то для всех $x \in R$ имеет место $\sum u_i q_i(x) = 0$. еорема доказана.

Следствие. Если система $f_i(x) < b_i$ неразрешима и ля всех $x \in R$ выполняется $f_r(x) = b_r$, то числа u_i, опре еленные в теореме 11, можно выбрать так, что $u_r > 0$. то вытекает из следствия теоремы 3 п. 2.2.

Мы доказали таким образом, что если не существует x, довлетворяющего неравенствам $f_i(x) < b_i$ для всех i, то нешние нормали $q_i(x)$ в произвольной точке x, удовлетво яющей системе $f_i(x) \leqslant b_i$, линейно зависимы с неотрица ельными коэффициентами, не зависящими от x.

Примером такой системы служит следующая система: $^2 + y < 0$, $x^2 - y < 0$, $z < 1$. Не существует действитель ых x, y, z, удовлетворяющих этой системе. Множеством R вляется луч $x=0$, $y=0$, $z \leqslant 1$. Мы имеем $q_1^T = (2x, 1, 0)$; $_2^T = (2x, -1, 0)$; $q_3^T = (0, 0, 1)$. Для всех $x \in R$ выпол яется равенство $q_1 + q_2 = 0$.

Теоремы 1—5 и 7—11 могут быть доказаны для вогну ых функций.

.5. Задача линейного программирования.

Пусть A — матрица $m \times n$ со строками $a_{i\cdot}^T$, $i \in I = \{1, \ldots, m\}$, столбцами $a_{\cdot j}$, $j \in J = \{1, \ldots, n\}$, и элемен ами a_{ij}; x и p — n-мерные векторы; y и b — m-мерные екторы. Сформулируем теперь задачу линейного програм ирования следующим образом:

$$\max \{p^T x \mid Ax \leqslant b, \quad x \geqslant 0\}. \tag{2.5.1}$$

аким образом, среди векторов, удовлетворяющих системе инейных неравенств $Ax \leqslant b$, $x \geqslant 0$, необходимо найти

вектор (векторы), на котором (на которых) линейная функ[...] называемая оптимизируемой (целевой), достигает максиму[...] или показать, что либо система $Ax \leqslant b$, $x \geqslant 0$, несовме[...] (в этом случае задача неразрешима), либо не существ[...] такого x', $Ax' \leqslant b$, $x' \geqslant 0$, что $p^T x \leqslant p^T x'$ для всех удовлетворяющих $Ax \leqslant b$, $x \geqslant 0$. В этом случае гово[...] что максимум бесконечен.

Ясно, что любую задачу линейного программирова[...] в том числе задачу с ограничениями в виде равенств[...] задачу с переменными, не ограниченными требованием неот[...]цательности, можно представить в форме (2.5.1). Вместо $Ax \leqslant$[...] мы будем иногда писать $Ax + y = b$, $y \geqslant 0$, или $\bar{A}\bar{x} =$[...] где

$$\bar{A} = (A, E); \quad \bar{x} = \begin{pmatrix} x \\ y \end{pmatrix} \qquad (2.5[...]$$

(E — единичная матрица порядка m).

Дадим следующие определения:

Допустимая область (*допустимое множеств*[...]) R — выпуклый многогранник [1]) в n-мерном пространстве, [...] точки которого удовлетворяют условиям $Ax \leqslant b$, $x \geqslant$[...]

Допустимая точка, или вектор (*возможное ре[...]ние*) [2]), — произвольный вектор $x \in R$, или вектор, удовлет[...]ряющий ограничениям. Если существует хотя бы один так[...] вектор, задача называется разрешимой (допустимой).

Базисное возможное решение [2]), *или вершина*, — прои[...]вольный вектор $x \in R$, который нельзя представить в ви[...] выпуклой линейной комбинации двух других допустим[...] векторов.

Оптимальная точка, или оптимальное решение [2]), [...] вектор x, являющийся решением задачи (2.5.1).

Для каждого $x \in R$ можно определить такие множест[...] $I(x) \subset I$ и $J(x) \subset J$, что $a_i^T . x = b_i$ при $i \in I(x)$, $a_i^T . x <$[...] при $i \in I - I(x)$, $x_j = 0$ при $j \in J(x)$, $x_j > 0$ при $j \in J - J(x$[...] Множество $I(x)$ ($J(x)$) может быть пустым и равным $I($[...]

[1]) Точнее, выпуклое многогранное множество. — *Прим. р*[...]

[2]) В отечественной литературе по линейному программирован[...] вместо терминов „возможное решение", „базисное возможное реш[...]ние" и „оптимальное решение" используются термины „план[...] „опорный план" и „оптимальный план" или „решение". — *Прим. р*[...]

Направление s в точке x называется *возможным*, если условия задачи не нарушаются при достаточно малом перемещении из x в направлении s, т. е.

$$\exists_\lambda \lambda > 0, \ x + \lambda s \in R. \qquad (2.5.3)$$

Для того чтобы направление s в точке x было возможным, необходимо и достаточно, чтобы выполнялись соотношения

$$a_i^T . s \leqslant 0 \quad \text{при} \quad i \in I(x),$$
$$s_j \geqslant 0 \quad \text{при} \quad j \in J(x). \qquad (2.5.4)$$

Возможное направление называется *подходящим*, если, кроме того,

$$p^T s > 0. \qquad (2.5.5)$$

Точка $x \in R$ — оптимальное решение, если в x не существует подходящего возможного направления, т. е. если $p^T s \leqslant 0$ для всех векторов s, удовлетворяющих (2.5.4).

Докажем теперь следующие теоремы.

Теорема 1. *Точка $x \in R$ оптимальна тогда и только тогда, когда целевой вектор p можно представить в виде линейной комбинации с неотрицательными коэффициентами внешних нормалей к граничным гиперплоскостям множества R, проходящим через точку x, т. е. когда существуют такие неотрицательные числа u_i и v_j ($i \in I(x)$, $j \in J(x)$), что*

$$p = \sum_{i \in I(x)} u_i a_i . - \sum_{j \in J(x)} v_j e_j.$$

Здесь e_j — единичный вектор с единицей на j-м месте.

Доказательство. Пусть x — оптимальная точка, тогда $p^T s \leqslant 0$ для любого s, удовлетворяющего неравенствам $a_i^T . s \leqslant 0$ при $i \in I(x)$, $-s_j \leqslant 0$ при $j \in J(x)$. Применяя теорему Фаркаша (теорема 1 п. 2.2), немедленно получаем требуемый результат. Обратное очевидно. Итак, в оптимальной точке x, y выполняются условия

$$p = A^T u - v, \ u \geqslant 0, \ v \geqslant 0,$$
$$u^T y = 0, \quad v^T x = 0. \qquad (2.5.6)$$

Эти условия можно получить, полагая $u_i = 0$ при $i \in I - I$
т. е. когда $y_i > 0$, и $v_j = 0$ при $j \in J - J(x)$, т. е. когда x_j

Рассмотрим теперь задачу линейного программиров

$$\min \{b^T u \mid A^T u \geqslant p, \ u \geqslant 0\}, \quad (2$$

получаемую из (2.5.1) с помощью следующих операций:

1) перестановки векторов p и b (т. е. целевого век и вектора ограничений),
2) транспонирования матрицы A,
3) изменения знака неравенства,
4) замены максимума на минимум.

Задача (2.5.7) называется *двойственной* по отноше к прямой задаче (2.5.1). Вместо $A^T u \geqslant p$ можно та писать $A^T u - v = p$, $v \geqslant 0$.

Теорема 2. *Прямая задача является двойствен по отношению к своей двойственной.*

Доказательство немедленно следует из правил 1—4.

Теорема 3. *Если* $\binom{x}{y}$ *и* $\binom{u}{v}$ *— возможные решен соответственно прямой и двойственной задач,* $p^T x \leqslant b^T u$.

Доказательство. $p^T x = u^T A x - v^T x = u^T b - u^T y - v^T x \leqslant b^T u$.

Теорема 4. *Если* $\binom{x'}{y'}$ *— оптимальное возможн решение прямой задачи, то двойственная задача доп стима и имеет ограниченное оптимальное решение. Для любого ее оптимального решения выполняютс соотношения*

$$p^T x' = b^T u'; \quad u'^T y' = 0, \quad v'^T x' = 0.$$

Доказательство. Пусть $\binom{x'}{y'}$ — оптимальное реш ние (2.5.1). Тогда из (2.5.6) следует существование возмож ного решения $\binom{u'}{v'}$ задачи (2.5.7), удовлетворяющего усл виям $u'^T y' = 0$ и $v'^T x' = 0$. При этом из доказательст

еоремы 3 вытекает, что $p^T x' = b^T u'$, т. е. $\binom{u'}{v'}$ — оптималь-ое решение. Из этого доказательства следует также, что оотношения $u^T y' = 0$, $v^T x' = 0$ выполняются для произволь-ого оптимального решения двойственной задачи.

Теорема 5. *Если* $\binom{x}{y}$ *и* $\binom{u}{v}$ *— возможные решения оответственно прямой и двойственной задач и* $u^T y = 0$, $^T x = 0$, *то эти решения оптимальны.*

Доказательство немедленно следует из теоремы 3 и ее оказательства.

Принимая во внимание, что каждое уравнение можно редставить в виде двух неравенств и каждую свободную еременную — как разность двух ограниченных, легко дока-ать следующие теоремы (штрих указывает на оптимальность оответствующих решений).

Теорема 6. *Прямой задаче* $\max \{p^T x \mid Ax = b, \ x \geqslant 0\}$ *оответствует двойственная задача* $\min \{b^T u \mid A^T u - v = p, \ v \geqslant 0\}$. *Для этой пары задач выполняются войственные соотношения*

$$p^T x' = b^T u'; \quad v'^T x' = 0.$$

Теорема 7. *Прямой задаче*

$$\max \{p^T x + q^T y \mid Ax + y = b, \ x \geqslant 0, \ y \geqslant 0\}$$

оответствует двойственная задача

$$\min \{b^T u \mid A^T u - v = p, \ u \geqslant q, \ v \geqslant 0\}.$$

Для этой пары задач выполняются двойственные оотношения

$$p^T x' + q^T y' = b^T u',$$
$$v'^T x' = 0, \quad u'^T y' = q^T y'.$$

Теорема 8. *Прямой задаче*

$$\max \{p_1^T x_1 + p_2^T x_2 \mid A_{11} x_1 + A_{12} x_2 + y_1 = b_1, \ y_1 \geqslant 0,$$
$$A_{21} x_1 + A_{22} x_2 = b_2, \ x_1 \geqslant 0\}$$

соответствует двойственная задача

$$\min \{b_1^T u_1 + b_2^T u_2 \mid A_{11}^T u_1 + A_{21}^T u_2 - v_1 = p_1, \; v_1 \geqslant 0,$$
$$A_{12}^T u_1 + A_{22}^T u_2 = p_2, \; u_1 \geqslant$$

Для этой пары задач выполняются двойственные со-ношения

$$p_1^T x_1' + p_2^T x_2' = b_1^T u_1' + b_2^T u_2'; \quad {v_1'}^T x_1' = 0; \quad {u_1'}^T y_1' = 0.$$

Таким образом, двойственные переменные, соответст-щие равенствам в прямой задаче, свободны, равен в двойственной задаче соответствуют свободным п менным прямой задачи. Всегда выполняются следу соотношения:

1. Оптимальные значения целевых функций прямо двойственной задач одинаковы.

2. Значения ограниченных переменных в оптимал решениях удовлетворяют так называемым соотноше дополняющей нежесткости [1]).

2.6. Задача выпуклого программирования.

Пусть $f_i(x)$, $i \in I_1 = \{1, \ldots, m_1\}$, — нелинейная выпу функция $x \in E_n$ с непрерывными частными производн a_i, $i \in I_2 = \{m_1 + 1, \ldots, m\}$, — n-мерный вектор; b_i, $i \in$ $= I_1 + I_2$, — действительные числа. Пусть, далее, $J = \{1, \ldots$ Определим выпуклое множество

$$R = \{x \mid f_i(x) \leqslant b_i \quad i \in I_1; \quad a_i^T x \leqslant b_i, \; i \in I_2, \; x \geqslant 0\}. \quad (2.$$

R называется допустимой областью, произвольный век $x \in R$ называется допустимым вектором или допусти точкой (возможным решением). В дальнейшем будем ис использовать соотношения

$$y_i = b_i - f_i(x) \geqslant 0, \quad i \in I_1,$$
$$y_i = b_i - a_i^T x \geqslant 0, \quad i \in I_2. \quad (2.$$

[1]) Например, для задач (2.5.1), (2.5.7) эти соотношения имеют

$$u^T y = 0, \quad v^T x = 0.$$

— *Прим. ред.*

радиент функции $f_i(x)$ будем обозначать через $q_i(x)$,

$$q_i(x) = \left\{ \frac{\partial f_i}{\partial x_j}, \quad j = 1, \ldots, n \right\}. \quad (2.6.3)$$

сли $f_i(x) = b_i$ для некоторых x и i, то $q_i(x)$ — внешняя ормаль к гиперповерхности $f_i(x) = b_i$ в точке x.

Наложим на R следующее *условие регулярности*:

$$1: \quad \forall_i (i \in I_1) \, \exists_x x \in R, \quad f_i(x) < b_i.$$

Теорема 1. *Если выполняется условие* C1, *то* $_i(x) < b_i$ *для некоторого* $x \in R$ *и всех* $i \in I_1$, *т. е.*

$$\exists_x x \in R, \quad \forall_i (i \in I_1) \, f_i(x) < b_i.$$

Доказательство. Для каждого $i \in I_1$ выберем $x^i \in R$ чтобы $f_i(x^i) < b_i$. Возьмем $x = \frac{1}{m_1} \sum_{h=1}^{m_1} x^h$, тогда, согласно еорема 2 п. 2.4,

$$f_i\left(\frac{1}{m_1} \sum_{h=1}^{m_1} x^h\right) \leqslant \frac{1}{m_1} \sum_{h=1}^{m_1} f_i(x^h) < b_i,$$

оскольку $f_i(x^h) \leqslant b_i$ для всех h и строго меньше b_i, огда $h = i$.

Для $x \in R$ положим $I_1(x) = \{i \mid f_i(x) = b_i\}$, $I_2(x) =$ $= \{i \mid a_i^T x = b_i\}$ и $J(x) = \{j \mid x_j = 0\}$.

Теорема 2. *Если выполняется условие* C1, $x \in R$ *и*

$$\sum_{\in I_1(x)} u_i q_i(x) + \sum_{i \in I_2(x)} u_i a_i - \sum_{j \in J(x)} v_j e_j = 0, \quad u_i \geqslant 0, \quad v_j \geqslant 0,$$

о $u_i = 0$ *для* $i \in I_1(x)$. *Это значит, что ни в одной очке* $x \in R$ *не существует внешних нормалей к нели-ейным гиперповерхностям* $f_i(x) = b_i$, *являющихся линей-ыми комбинациями с неположительными коэффициен-ами других внешних нормалей в той же точке.*

Доказательство. Предположим, что $u_i > 0$ для екоторого $i \in I_1(x)$, например $u_r > 0$. Выпуклая функция

$\sum_{i\in I_1(x)} u_i f_i(x) + \sum_{i\in I_2(x)} u_i a_i^T x - \sum_{j\in J(x)} v_j x_j$ [1] равна $\sum_{i\in I(x)}$ во всех точках $x\in R$ $(I(x)=I_1(x)+I_2(x))$. Следователь неравенство $f_r(x)<b_r$ не может иметь места в точках $x\in$ что противоречит условию C1.

Следствие. Если выполнено условие C1, $x\in R$ $i\in I_1(x_1)$, то $q_i(x)\neq 0$.

Пусть $x\in R$. Направление s в точке x называется *возмо ным*, если достаточно малое перемещение из x в направ нии s не выводит за пределы области R, т. е. если $\exists_\lambda \lambda >$ $x+\lambda s\in R$. Рассмотрим $S(x)$ — многогранный конус, сос вленный из всех векторов s, удовлетворяющих услови $q_i(x)^T s\leqslant 0$ при $i\in I_1(x)$, $a_i^T s\leqslant 0$ при $i\in I_2(x)$ и $s_j\geqslant$ при $j\in J(x)$. Для того чтобы направление s было возможн необходимо, чтобы $s\in S(x)$. Если $I_1(x)$ непусто, это усло не является, вообще говоря, достаточным.

В литературе вместо нашего условия C1 обычно на накладывается условие C2: для любой точки $x\in R$ про вольный вектор $s\in S(x)$ касается некоторой дуги, целик содержащейся в допустимой области (см. Кун и Таккер [и Деннис [19, стр. 161]). Условие C2, как легко доказа эквивалентно условию, что для всех $x\in R$ замыкание кон возможных направлений $S'(x)$ совпадает с $S(x)$. Как ви из теоремы 3, ограничения, накладываемые условием (слабее, чем ограничения, накладываемые условием C1.

Теорема 3. *Если выполняется условие* C1, *то вып няется и* C2.

Доказательство. Предположим, что выполняет условие C1, так что для некоторого $\bar{x}\in R$ и всех $i\in$ $f_i(\bar{x})<b_i$. Возьмем произвольную точку $x\in R$ и рассмотр возможное направление $\bar{s}=\bar{x}-x$. Согласно следствию теоремы 3 п. 2.4, $q_i(x)^T\bar{s}<0$ для всех $i\in I_1(x)$. Если $s\in S($ то для любого $\lambda>0$ из этого следует, что $q_i(x)^T(s+\lambda\bar{s})<$

[1] $I_1(x)$, $I_2(x)$, $J(x)$ определены здесь для фиксированного x. — *Прим. перев.*

$i\in I_1(x)$. Тогда $s+\lambda\bar{s}\in S'(x)$ для всех $\lambda>0$. Это означа т выполнение условия C2.

Таким образом, условия C2 выполняются для более широ о класса ограничений, чем C1, однако примеры, когда C2 олняется, а C1 нет, в известной степени искусственны. тих примерах $f_i(x)=b_i$ для всех $x\in R$ и некоторых i, ример $i\in I_1'\subset I_1$. Ни C1, ни C2 не представляются суще енными ограничениями на R. Заметим, что если условие C1 выполняется, то замена всех ограничений $f_i(x)\leqslant b_i$ огра ениями $f_i(x)\leqslant b_i+\varepsilon$, где ε — малое положительное число, ет к допустимой области $R(\varepsilon)\supset R$, для которой усло C1 выполняется (в предположении, что R непусто). ло ε может добавляться только к тем ограничениям $x)\leqslant b_i$, для которых неравенства $f_i(x)<b_i$ не выпол ся ни в одной точке $x\in R$.

Пусть $F(x)$ — вогнутая дифференцируемая функция с не рывным градиентом; множество R, определяемое форму й (2.6.1), удовлетворяет условию C1. Тогда задача

$$\max\{F(x)\mid x\in R\} \qquad (2.6.4)$$

вается *задачей выпуклого программирования*. Функция $F(x)$ называется *оптимизируемой*. Возможное равление s в точке x называется *подходящим*, если

$$\left(\frac{\partial F(x+\lambda s)}{\partial\lambda}\right)_{\lambda=0}=g(x)^T s>0.$$

Теорема 4. *Если в точке $x\in R$ не существует под дящего возможного направления, то функция $F(x)$ этой точке достигает своего максимума на R.*

Доказательство. Предположим, что существует ка $y\in R$, для которой $F(y)>F(x)$. Функция $-F(x)$ пуклая, поэтому, согласно следствию 1 теоремы 3 п. 2.4, $x)^T(y-x)>0$. Поскольку направление $y-x$ возможное, оно подходящее. Таким образом, если подходящих воз жных направлений не существует, то $F(x)$ достигает точке x максимума на R.

Теорема 5. *Для того чтобы функция $F(x)$ достигала ксимума на R в точке $x\in R$, необходимо и достаточно,*

чтобы $g(x)^T s \leqslant 0$ для всех s, удовлетворяющих
виям $q_i(x)^T s \leqslant 0$ для $i \in I_1(x)$, $a_i^T s \leqslant 0$ для $i \in$
и $s_j \geqslant 0$ для $j \in J(x)$, т. е. для всех $s \in S(x)$.

Доказательство. Достаточность сразу следу
определения подходящего направления и теоремы 4.
доказательства необходимости предположим, что сущес
вектор $s \in S(x)$, для которого $g(x)^T s > 0$. Из теоре
известно, что для некоторого x' и всех $i \in I_1$ выполн
неравенство $f_i(x') < b_i$. Положим $s' = x' - x$,
$q_i(x)^T s' < 0$ для всех $i \in I_1(x)$. Если $s(\lambda) = \lambda s + s'$
$s(\lambda)$ — возможное направление для всех $\lambda \geqslant 0$, поскольк
всех $\lambda \geqslant 0$ $q_i(x)^T s(\lambda) < 0$ при $i \in I_1(x)$, $a_i^T s(\lambda) \leqslant 0$
$i \in I_2(x)$ и $\{s(\lambda)\}_j \geqslant 0$ при $j \in J(x)$. Так как $g(x)^T s$
можно выбрать столь большое λ, что $g(x)^T s(\lambda) > 0$. С
вательно, x не может быть максимальной точкой.

Замечание. Если потребовать, чтобы R удовлетво
условию C2, а не C1, теорема остается справедливой. До
тельство приведено в [19, стр. 162—164].

Теорема 6. *Точка $x \in R$ является точкой ма*
мума функции $F(x)$ на R тогда и только тогда, к
градиент в точке x можно представить как лине
комбинацию с неотрицательными коэффициент
внешних нормалей в x, т. е.

$$x\,\text{maximum} \rightleftarrows g(x) = \sum_{i \in I_1(x)} u_i q_i(x) + \sum_{i \in I_2(x)} u_i a_i -$$
$$- \sum_{j \in J(x)} v_j e_j;\quad u_i \geqslant 0,\quad v_j \geqslant 0. \qquad (2$$

Доказательство. Если функция $g(x)$ может
представлена в виде (2.6.5), то $g(x)^T s \leqslant 0$ для лю
$s \in S(x)$ и, в силу теоремы 5, $F(x)$ достигает в точк
максимума на R. С другой стороны, если x — точка ма
мума $F(x)$ на R, то $g(x)^T s \leqslant 0$ для любого $s \in S(x)$, и п
ставление (2.6.5) следует из теоремы Фаркаша. Вместо (2.

ожно также написать

$$x\,\text{maximum} \rightleftarrows g(x) = \sum_{i \in I_1} u_i q_i(x) + \sum_{i \in I_2} u_i a_i - v;$$
$$u \geqslant 0,\ v \geqslant 0,\ u^T y = 0,\ v^T x = 0, \qquad (2.6.6)$$

е y определяется формулой (2.6.2).

Кун и Таккер [22] показали, что задача максимизации
гнутой дифференцируемой функции на выпуклом мно-
естве R, определяемом условиями $x \geqslant 0$, $f_i(x) \leqslant b_i$,
$\in I$ [$f_i(x)$—выпуклые дифференцируемые функции] и удовле-
воряющем условию C2, эквивалентна задаче определения
дловой точки функции $\varphi(x, u) = F(x) + u^T y$ при усло-
иях $u \geqslant 0$, $x \geqslant 0$, где $y_i = b_i - f_i(x) \geqslant 0$. При этом
седловой точке x', u' выполняются следующие условия:

1. $\left(\frac{\partial \varphi(x, u)}{\partial x}\right)_{x', u'} \leqslant 0$, т. е. $g(x') - \sum u_i' q_i(x') \leqslant 0$,

$x'^T \left(\frac{\partial \varphi(x, u)}{\partial x}\right)_{x', u'} = 0$, т. е. если $x_j' > 0$, то

$g_j(x') - \sum u_i' \{q_i(x')\}_j = 0$.

2. $\left(\frac{\partial \varphi(x, u)}{\partial u}\right)_{x', u'} \geqslant 0$, т. е. $f_i(x') - b_i \leqslant 0$,

$u'^T \left(\frac{\partial \varphi(x, u)}{\partial u}\right)_{x', u'} = 0$, т. е. если $f_i(x') < b_i$, то $u_i' = 0$.

ги условия называются условиями оптимальности Куна —
ккера. Они совпадают с условиями (2.6.6).

Теорема 7. *Пусть x' — точка максимума функ-*
ии $F(x)$ на множестве R. Тогда x' является также
ешением задач:

1. $\max \{g(x')^T x \mid x \in R\}$.
2. $\max \{g(x')^T x \mid q_i(x')^T x \leqslant q_i(x')^T x',\ i \in I_1(x');\ a_i^T x \leqslant b_i,$
$\in I_2(x');\ x_j \geqslant 0,\ j \in J(x')\}$.

Доказательство. Так как x' — точка максимума
(x) на R, то в ней выполняются условия (2.6.6). В зада-
х 1 и 2 точка x' допустимая, и в ней выполняются усло-
я оптимальности, совпадающие с условиями (2.6.6), поэтому
— решение задач 1 и 2. Заметим, что задача 2 является
дачей линейного программирования.

Теорема 8. *Для любого $\bar{x} \in R$ справедливы н*
венства

$$\max\{F(x) \mid x \in R\} - F(\bar{x}) \leqslant \max\{g(\bar{x})^T x \mid x \in R\} -$$
$$- g(\bar{x})^T \bar{x} \leqslant \max\{g(\bar{x})^T x \mid q_i(\bar{x})^T x \leqslant q_i(\bar{x})^T \bar{x} + b_i - f_i$$
$$i \in I_1; \quad a_i^T x \leqslant b_i, \quad i \in I_2; \quad x \geqslant 0\} - g(\bar{x})^T \bar{x}. \quad (2$$

Доказательство. $F(x)$ вогнута, т. е., согласно
реме 3 п. 2.4 [$-F(x)$ — выпуклая функция], для люб
$x \in R$

$$F(x) - F(\bar{x}) \leqslant g(\bar{x})^T (x - \bar{x}) \leqslant$$
$$\leqslant \max\{g(\bar{x})^T x \mid x \in R\} - g(\bar{x}$$

Отсюда следует первое неравенство. Пусть $x \in R$,
$f_i(x) \leqslant b_i$ для всех i. Поскольку функции $f_i(x)$ выпу
из теоремы 3 п. 2.4 вытекает, что

$$f_i(x) \geqslant f_i(\bar{x}) + q_i(\bar{x})^T (x - \bar{x}),$$

т. е.

$$q_i(\bar{x})^T x \leqslant q_i(\bar{x})^T \bar{x} + b_i - f_i(\bar{x}).$$

Таким образом, выпуклый многогранник [1]), определя
этими неравенствами, содержит R. Отсюда следует вт
неравенство.

Теорему 8 можно рассматривать как теорему об ап
ксимации. Если мы начинаем итерационное решение в д
стимой точке $\bar{x}$, то максимальное приращение оптимиз
мой функции не может быть больше приращения опти
руемой функции в линеаризованной задаче.

Линеаризация состоит в замене максимизации $F(x)$
симизацией линейных членов разложения $F(x)$ в ряд
лора и в замене ограничений $y_i = b_i - f_i(x) \geqslant 0$ линей
неравенствами, получаемыми разложением $b_i - f_i(x)$
Тейлора и отбрасыванием всех членов, кроме линейны

Если $\bar{x}$ в действительности является максимумом, то
равенства (2.6.7) переходят в равенства, поэтому под
аппроксимация приемлема.

[1]) Точнее, выпуклое многогранное множество. — *Прим. р*

Глава 3

МЕТОДЫ РЕШЕНИЯ ЗАДАЧ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

.1. Введение.

В настоящее время известно много методов решения за-
ач линейного программирования. По числу итераций их
ожно подразделить на конечные и бесконечные. Первая
руппа обычно основывается на методе исключения Гаусса.
 ней относятся симплексный, двойственный симплексный ме-
оды, метод ведущих переменных Била [3], метод одновремен-
ого *решения прямой и двойственной задач и методы возмож-*
ых направлений [1]). Первый, второй и четвертый методы опи-
аны в настоящей главе, последние рассматриваются в гл. 9.

Группа бесконечных методов включает все виды методов
елаксации. Несмотря на незначительный объем вычислений
а каждой итерации, эти методы обычно требуют такого
ольшого числа шагов, т. е. сходятся так плохо, что их
пособность конкурировать с конечными методами для задач
ольших размеров сомнительна.

.2. Симплексный метод.

Симплексный метод является наиболее широко известным
исленным методом решения задач линейного программиро-
ания. Он описан во многих статьях и книгах, например
Данцигом, Орденом и Вулфом [17], Вайда [8], Гассом [10].

Для изложения симплексного метода предположим прежде
всего, что задача поставлена в следующей форме (обозна-
чения см. в п. 2.5):

$$\max\{p^T x \mid Ax + y = b, \ x \geqslant 0, \ y \geqslant 0, \ b \geqslant 0\}. \quad (3.2.1)$$

[1]) В отечественной литературе симплексный метод, двойственный симплексный метод и метод одновременного решения прямой *и двойственной задач называются соответственно методом последо*вательного улучшения плана, методом последовательного уточнения оценок и методом последовательного сокращения невязок. — *Прим. ред.*

Переменные y_i будем называть *дополнительными*. Обо чая $\bar A=(A, E)$, $\bar x = \begin{pmatrix} x \\ y \end{pmatrix}$ и $\bar p^T=(p^T, 0)$, приведем за (3.2.1) к виду

$$\max\{\bar p^T\bar x \mid \bar A\bar x = b \geqslant 0,\ \bar x\geqslant 0\}. \quad (3$$

Предположим на время, что выполняется условие невырож ности: для любой неособенной подматрицы B порядк матрицы (A, E) из $B^{-1}b\geqslant 0$ следует $B^{-1}b>0$ (т. компоненты $B^{-1}b$ положительны). Можно показать, что метрически это требование равносильно условию, согл которому каждая вершина допустимой области не принад жит ни одной граничной гиперплоскости, кроме опреде щих ее n линейно независимых граничных гиперплоскостей требование означает также, что любое возможное реш содержит по меньшей мере m положительных компон Ясно, что в невырожденной задаче $b>0$.

Будем считать известной неособенную подматр $B=(\bar a_{\cdot j_1},\ \ldots,\ \bar a_{\cdot j_m})$ порядка m, удовлетворяющую усло $B^{-1}b>0$ ($\bar a_{\cdot j}$ — j-й столбец матрицы $\bar A$, $j=1,\ \ldots,\ m+$ $\bar a_{\cdot n+r}=e_r$ — r-й единичный вектор). Представим $\bar A$, $\bar p$ в виде $\bar A=(B, D)$, $\bar p=\begin{pmatrix}\bar p_B \\ \bar p_D\end{pmatrix}$, $\bar x=\begin{pmatrix}\bar x_B \\ \bar x_D\end{pmatrix}$, тогда $b=B\bar x_B+D$

т. е. $\bar x_B = B^{-1}b - B^{-1}D\bar x_D$. Отсюда следует, что $\bar x_B = B^{-1}b$ $\bar x_D=0$ — базисное возможное решение со значением о мизируемой функции $\bar p_B^T B^{-1}b$. Матрицу B будем назы *базисом*. Переменные $\bar x_{B_i}=\bar x_{j_i}$ $(i=1,\ \ldots,\ m)$ назыв *базисными*, а $\bar x_{Dj}$ — *внебазисными* переменными. Подс вляя выражение x_B в оптимизируемую функцию, полу

$$\bar p^T\bar x = \bar p_B^T\bar x_B + \bar p_D^T\bar x_D = \bar p_B^T B^{-1}b - (\bar p_B^T B^{-1}D - \bar p_D^T)\,\bar x_D.$$

Определим вектор $\bar d$, компоненты которого

$$\bar d_j = \bar p_B^T B^{-1}\bar a_{\cdot j} - \bar p_j,\qquad j=1,\ \ldots,\ m+n. \quad (3.2$$

Таким образом, $\bar d_j=0$, если $\bar x_j$ — базисная переменная. Пр ставляя $\bar d$ в виде $\bar d^T=(d_1^T,\ d_2^T)$, где d_1 — n-мерный век соответствующий переменным x_j, а d_2 — m-мерный век

ответствующий переменным y_i, получаем

$$d_2^T=\bar p_B^T B^{-1};\quad d_1^T=\bar p_B^T B^{-1}A - p^T. \quad (3.2.4)$$

еличины $\bar d_j$ часто называют относительными оценками. троку относительных оценок будем называть d-строкой. з изложенного следует, что

$$\bar p^T\bar x = \bar p_B^T B^{-1}b - \bar d^T\bar x.$$

огут иметь место два случая:

1. $\bar d_j\geqslant 0$ для всех j. Тогда $\bar x=\begin{pmatrix}\bar x_B \\ \bar x_D\end{pmatrix}=\begin{pmatrix}B^{-1}b \\ 0\end{pmatrix}$ — опти льное решение. Действительно, векторы $u^T=d_2^T$ и $v^T=d_1^T$ овлетворяют ограничениям задачи, двойственной к (3.2.1), то время как $u^Ty=0$ и $v^Tx=0$, так что оптимальность едует из теоремы 5 п. 2.5.

2. $\exists_s\bar d_s<0$. Будем теперь увеличивать $\bar x_s$ от нуля до тех р, пока одна из ненулевых (базисных) переменных $\bar x_{B_l}$ не ратится в нуль. Мы должны, следовательно, определить аляр $\lambda_r>0$, такой, что

$$\lambda_r=\max\{\lambda\mid B^{-1}b-\lambda B^{-1}\bar a_{\cdot s}\geqslant 0\}. \quad (3.2.5)$$

сли $\lambda_r=+\infty$, т. е. $B^{-1}\bar a_{\cdot s}\leqslant 0$, то решение задачи не раничено. Действительно, точка с координатами $\bar x_s=\lambda$, $\bar x_B=B^{-1}b-\lambda B^{-1}\bar a_{\cdot s}$, $\bar x_{D_j}=0$ при $\bar x_{D_j}\neq\bar x_s$ является воз можным решением при любом $\lambda>0$, и в то же время при λ, тремящемся к ∞, значение $\bar p^T\bar x=\bar p_B^TB^{-1}b-\lambda\bar d_s$ превосхо ит любое наперед заданное число.

Если $\lambda_r<\infty$, то r определяется единственным образом, ак как в противном случае $\bar x_s=\lambda_r$, $\bar x_B=B^{-1}b-\lambda_rB^{-1}\bar a_{\cdot s}$ $\bar x_{D_j}=0$ при $\bar x_{D_j}\neq\bar x_S$ — возможное решение с числом по ожительных компонент, меньшим m, что противоречит пред оложению о невырожденности. Теперь можно разрешить -е уравнение относительно $\bar x_S$ [очевидно, $(B^{-1}\bar a_{\cdot s})_r>0$] и сключить $\bar x_S$ из остальных уравнений и оптимизируемой функции. Так мы получим новые базисные переменные и

новый базис B^*. Введем *η-вектор* η_r, задавая его комп

$$(\eta_r)_i = -\frac{(B^{-1}\overline{a}_{\cdot s})_i}{(B^{-1}\overline{a}_{\cdot s})_r} \quad \text{при} \quad i \neq r,$$

$$(\eta_r)_r = \frac{1}{(B^{-1}\overline{a}_{\cdot s})_r}$$

и *элементарную матрицу*

$$E_r = (e_1, \ldots, e_{r-1}, \eta_r, e_{r+1}, \ldots, e_m).$$

Тогда

$$(B^*)^{-1} = E_r B^{-1},$$

и для любого столбца матрицы $\overline{A}$

$$(B^*)^{-1}\overline{a}_{\cdot j} = E_r(B^{-1}\overline{a}_{\cdot j}). \quad (3$$

Если мы располагаем таблицей со столбцами $B^{-1}\overline{a}$ можно получить новую таблицу, соответствующую н базису, умножая каждый столбец старой на элемент матрицу E_r. При этом единичный столбец $B^{-1}\overline{a}_{\cdot j_r}$ пре зуется в столбец η_r, а столбец $B^{-1}\overline{a}_{\cdot s}$ преобразуется в ничный. Умножение на E_r приводит к хорошо извес формулам, аналогичным тем, которые используются в м Жордана для обращения матриц.

Столбец, вводимый в базис на текущей итерации, вается *главным* или *направляющим*. Индекс r, задав соотношением (3.2.5), определяет *главную*, или *напра щую*, строку. Общий элемент $(B^{-1}\overline{a}_{\cdot s})_r$ главного ст и главной строки называется *направляющим*.

При указанном изменении базиса значение оптимизир функции возрастает на величину $-\lambda_r \overline{d}_s > 0$.

Действуя и дальше таким образом, получим после тельность базисов и соответствующих им базисных ре с монотонно возрастающими значениями оптимизир функции. Поскольку число возможных базисов конеч ни один базис не встречается дважды, если задача пре гается невырожденной, решение заканчивается за ко число шагов.

Из предыдущего следует, что задача решена, как т найдена квадратная подматрица B' матрицы $\overline{A}$, облад свойствами:

1. $$\{B'\}^{-1}b \geqslant 0 \quad (3$$

(в предположении о невырожденности выполняется строгое неравенство).

2. $$\overline{d}'_j = \overline{p}^T_{B'}\{B'\}^{-1}\overline{a}_{\cdot j} - \overline{p}_j \geqslant 0 \text{ для всех } j = 1, \ldots, m+n. \quad (3.2.12)$$

При этом оптимальным решением будет

$$\overline{x}_{B'} = \{B'\}^{-1}b, \quad \overline{x}_{D'} = 0. \quad (3.2.13)$$

Прямой подстановкой в формулы (2.5.7) можно проверить, что равенства

$$u'^T = \overline{p}^T_{B'}\{B'\}^{-1} = d_2'^T, \quad v'^T = \overline{p}^T_{B'}\{B'\}^{-1}A - p^T = d_1'^T \quad (3.2.14)$$

определяют оптимальное решение двойственной задачи. Это решение получается одновременно с решением прямой задачи посредством вектора d.

Подводя итоги, можно сказать, что ν-я итерация симплексного метода состоит из следующих операций.

1. Обычно выбирается (если существует) наименьшее значение $d_{s_\nu}^{\nu-1} < 0$ и определяется номер главного столбца. Здесь

$$(\overline{d}^{\nu-1})^T = \begin{pmatrix} d_1^{\nu-1} \\ d_2^{\nu-1} \end{pmatrix}^T = \left(\overline{p}^T_{B^{\nu-1}}\{B^{\nu-1}\}^{-1}A - p^T,\ \overline{p}^T_{B^{\nu-1}}\{B^{\nu-1}\}^{-1}\right),$$

$B^{\nu-1}$ — подматрица порядка m матрицы (A, E), составленная из столбцов, соответствующих базисным переменным $(\nu - 1)$-й итерации.

2. Для выбранного направляющего столбца номер r_ν главной строки определяется условиями

$$\frac{b_{r_\nu}^{\nu-1}}{\overline{a}_{r_\nu s_\nu}^{\nu-1}} = \min_i \left\{ \frac{b_i^{\nu-1}}{\overline{a}_{is_\nu}^{\nu-1}} \,\middle|\, \overline{a}_{is_\nu}^{\nu-1} > 0 \right\}, \quad (3.2.15)$$

где $b^{\nu-1} = \{B^{\nu-1}\}^{-1}b$, $a_{\cdot s_\nu}^{\nu-1} = \{B^{\nu-1}\}^{-1}\overline{a}_{\cdot s_\nu}$.

3. Производится построение нового η-вектора:

$$\{\eta_{r_\nu}^\nu\}_i = -\frac{\overline{a}_{is_\nu}^{\nu-1}}{\overline{a}_{r_\nu s_\nu}^{\nu-1}}, \quad i \neq r_\nu, \quad (3.2.16)$$

$$\{\eta_{r_\nu}^\nu\}_{r_\nu} = \frac{1}{\overline{a}_{r_\nu s_\nu}^{\nu-1}} \quad (3.2.17)$$

и выполняется преобразование

$$b^{\nu} = E^{\nu}_{r_\nu} b^{\nu-1}, \qquad (3.2$$

$$\bar{a}_{\cdot j}^{\nu} = E^{\nu}_{r_\nu} \bar{a}_{\cdot j}^{\nu-1}, \qquad j = 1, \ldots, n+m. \qquad (3.2$$

Отметим, что в процессе отдельной итерации прове всех этих вычислений не обязательно.

Вычислительные алгоритмы решения будут исследов в гл. 4 и 5.

До сих пор предполагалось, что условия задачи зада в виде (3.2.1), так что немедленно получается началь базис $B^0 = E$, $\bar{x}_B = y = b$, $\bar{x}_D = x = 0$, однако это не все так. Мы будем различать следующие случаи:

1. Оптимизируемая функция зависит от некоторых пе менных y_i, т. е. $\bar{p}^T = (p^T, q^T)$. В этом случае в качес исходной d-строки выберем

$$d_1^0 = A^T q - p, \quad d_2^0 = 0$$

и используем $-d^0$ в качестве оптимизируемой функции (так образом новый вектор $\bar{p} = -d^0$). Легко проверить (см. т рему 7 п. 2.5), что решением двойственной задачи бу

$$u' = d_2' + q, \quad v' = d_1'. \qquad (3.2.$$

2. $b \geqslant 0$, но матрица $\bar{A}$ содержит меньше чем m е ничных векторов. Если матрица $\bar{A}$ не содержит вектор e_i $(i \in I_z \subset I)$, то можно, например, добавить искусствен переменные z_i, $i \in I_z$, и соответствующие единичные вектор Решим прежде всего искусственную задачу (первый эт

$$\max\Big\{-\sum_{i \in I_z} z_i \,\Big|\, a_i^T.\, x + y_i = b_i \quad \text{при} \quad i \in I - I_z;$$

$$a_i^T.\, x + z_i = b_i \quad \text{при} \quad i \in I_z;\ x \geqslant 0,\ y_i \geqslant 0,\ z_i \geqslant 0,\ b \geqslant 0 \qquad (3.2.2$$

где вектор-строки $a_i^T.$ — строки матрицы A.

Мы получили задачу линейного программирования ти (3.2.1). Ее можно решить за конечное число шагов. Опти мальное значение может оказаться отрицательным или ра ным нулю. Если $\sum z_i > 0$, исходная система ограничен

несовместна, и задача неразрешима. Если $\sum z_i = 0$, получим базисное возможное решение в предположении невырожденности не содержащее базисных переменных z_i. Мы можем полностью забыть искусственные переменные и, отправляясь от построенного начального базиса B, перейти ко второму этапу — построению решения исходной задачи.

3. $b_i < 0$ для некоторых i. Если такая строка не содержит переменной y_i, то, умножив ее на -1, введем искусственную переменную z_i. Если переменная y_i содержится в строке, то после умножения этой строки на -1 y_i не может входить в начальный базис, так что и в этом случае необходимо введение искусственной переменной. В обоих случаях сначала приходится решать задачу типа (3.2.21).

4. Некоторые переменные x_j не ограничены требованием неотрицательности. Всегда можно полагать $x_j = x_j^+ - x_j^-$, $x_j^+ \geqslant 0$, $x_j^- \geqslant 0$ для каждой такой свободной переменной. Если $d(x_j) \leqslant 0$, в базис вводится x_j^+, в противном случае — x_j^-. После того как это проделано, соответствующая строка таблицы исключается из рассмотрения при выборе направляющей строки.

Следует иметь в виду, что описанные методы не всегда являются наиболее эффективными с точки зрения вычислений.

Доказательство конечности симплексного метода оказывается неверным в часто встречающемся случае вырожденности, так как вырожденность может привести к неединственности выбора λ_r. Интуитивно ясно, что небольшой сдвиг вектора ограничений b, сделанный подходящим образом, может обеспечить единственность. На этой идее основан первый из двух следующих способов преодоления вырожденности.

I способ преодоления вырожденности (см. Чарнс [36]):

а. Найдем $$I_0 = \left\{ r \in I \,\middle|\, \lambda_r = \frac{b_r^*}{\bar{a}_{rs}^*} = \min_i \frac{b_i^*}{\bar{a}_{is}^*}, \quad \bar{a}_{is}^* > 0 \right\}$$

[звездочкой отмечено преобразование; так, $b_i^* = (B^{-1}b)_i$, где B^{-1} — матрица, обратная текущему базису].

Если I_0 пусто, задача имеет неограниченное решение, если I_0 состоит из одного элемента, выбор λ_r^* определяется единственным образом.

b. Предположим, что множества $I_0, I_1, \ldots, I_{k-1}$ опр лены, и I_{k-1} состоит более чем из одного элемента. Опр лим теперь

$$I_k = \left\{ r \in I_{k-1} \,\middle|\, \frac{\bar a^*_{rk}}{\bar a^*_{rs}} = \min_i \frac{\bar a^*_{ik}}{\bar a^*_{is}}, \quad i \in I_{k-1} \right\}.$$

Так как строки матрицы $\bar A$ линейно независимы ($\bar A$ содер единичную матрицу), мы получим единственное r после нечного числа шагов, меньшего числа столбцов матр (возможно, расширенной введением искусственных вектор

II способ преодоления вырожденности (см. Данциг, Ор и Вулф [17]):

a. Найдем $I_0 \subset I$ как в I способе.

b. Предположим, что множества $I_0, I_1, \ldots, I_{k-1}$ опре лены. Определим теперь

$$I_k = \left\{ r \in I_{k-1} \,\middle|\, \frac{(B^{-1}e_k)_r}{\bar a^*_{rs}} = \min_i \frac{(B^{-1}e_k)_i}{\bar a^*_{is}}, \quad i \in I_{k-1} \right\}.$$

Так как строки обратной матрицы независимы, мы получ единственное r после конечного числа шагов, не превос дящего m.

Можно доказать, что оба эти способа гарантируют ре ние задачи линейного программирования за конечное чис шагов. Следовательно, и I этап решения задачи (3.2.21) кончится за конечное число шагов. Но в случае вырожден некоторые переменные z_i могут оставаться в базисе (с ну выми значениями, так как в противном случае задача нер решима). Мы можем, однако, перейти ко второму этап добавляя к ограничениям задачи уравнение $\sum z_i \leqslant 0$, суммирование производится по тем $i \in I_z$, для которых входят в конечный базис I этапа. Дополнительное ура нение гарантирует, что ни одно из этих z_i не станет сно положительным.

3.3. Двойственный симплексный метод.

Будем рассматривать задачу линейного программирован в форме (2.5.1), где $p_j \leqslant 0$ для всех j, $b_i < 0$ для некото рых i. Из (2.5.7) следует, что $u = 0$, $v = -p$ образую

ходное базисное решение двойственной задачи, так что эта дача может решаться непосредственно, без привлечения кусственных приемов. Используя симплексный метод для шения двойственной задачи, мы получим одновременно шение прямой задачи [см. теорему 2 п. 2.5 и (3.2.14)]. Однако вместо того, чтобы переходить к двойственной даче и затем решать ее симплексным методом, можно, как казал Лемке [23], оперировать с прямой задачей, используя итерии, ведущие к выбору тех же главных строк и столб в, которые выбирались бы при приложении симплексного тода к двойственной задаче.

Таким образом, мы получим двойственный симплексный тод, содержащий следующие основные операции:

1. На ν-й итерации направляющая строка (если она суще вует) определяется условием $b^{\nu-1}_{r_\nu} < 0$ (обычно выбирают року с наименьшим значением $b^{\nu-1}_{r_\nu}$). Здесь $b^{\nu-1} = \{B^{\nu-1}\}^{-1} b$, $^{-1}$ — подматрица порядка m матрицы (A, E), являющаяся зисом прямой задачи на $\nu - 1$-й итерации.

2. Для выбранной главной строки определим номер s_ν правляющего столбца:

$$\frac{\bar d^{\nu-1}_{s_\nu}}{-\bar a^{\nu-1}_{r_\nu s_\nu}} = \min_j \left\{ \frac{\bar d^{\nu-1}_j}{-\bar a^{\nu-1}_{r_\nu j}} \,\middle|\, \bar a^{\nu-1}_{r_\nu j} < 0 \right\} \qquad (3.3.1)$$

предположении, что $\bar a^{\nu-1}_{r_\nu j} < 0$ для некоторого j. В про вном случае задача неразрешима.

3. Формулы преобразования те же, что и в симплексном етоде.

4. Если $b^k_i \geqslant 0$ для всех i, то задача решена на k-й ите ации [условия (3.3.1) обеспечивают выполнение неравенств $_j \geqslant 0$ для всех j и ν].

Если принять меры преодоления вырожденности, решение остигается за конечное число шагов.

Легко дать геометрическую интерпретацию двойственного имплексного метода в пространстве, соответствующем пере енным прямой задачи. Любое неоптимальное базисное воз ожное решение двойственной задачи соответствует недо устимой вершине прямой задачи (т. е. пересечению n

линейно независимых граничных гиперплоскостей[1]). Эт является оптимальным решением ограничений прямой полученной исключением из ограниченной данной неудовлетворенных неравенств. Такую точку будем на недопустимой оптимальной базисной точкой. Решая двойственным симплексным методом, мы движемся от шины к вершине по недопустимым оптимальным ба точкам до тех пор, пока не будет достигнуто оптим базисное возможное решение.

Необходимо заметить, что многие вершины не яв ни возможными решениями прямой задачи, ни возмо решениями двойственной. Если начальная вершина я такой точкой, мы вынуждены использовать искусст приемы в обоих методах. В симплексном методе искус ные переменные добавляются к ограничениям, являю равенствами, и к ограничениям вида неравенств при затем решается задача (3.2.21) (I этап п. 3.2). В двой ном симплексном методе искусственные переменные ляются к ограничениям вида равенств и вводится до тельное ограничение $\sum x_j \leqslant M$; суммирование произ по всем j, для которых $p_j > 0$ ($j \in J_+$), а M столь в что это ограничение не влияет на оптимальное ре Выберем в качестве направляющей эту строку и стол определяемый условием $p_s = \max p_j$. Тогда $b_i' = b_i$ $i = 1, \ldots, m$, $b_{m+1}' = M$ — новый вектор огра а $p_j' = p_j - p_s$, когда $j \in J_+$, и $p_j' = p_j$, когда j $p_j' \leqslant 0$ для всех j. Вместо этого можно решать зад этапа. На первом этапе выбор главных строк опред столбцом $-a_{\cdot s}$. Первый этап продолжается до т пока элементы столбца s станут неотрицательными чего выбор направляющей строки определяется стол

Преимущество двойственного симплексного метода чается в том, что искусственные переменные можно цессе решения исключать одно за другим. Недост состоит в том, что мы не располагаем возможным ре

[1]) Недопустимая вершина образована пересечением независимых граничных гиперплоскостей многогранного прямой задачи, но не является вершиной этого мно *Прим. ред.*

самого конца вычислений, т. е. необходимо полностью шить задачу, прежде чем результаты будут иметь практи-кий смысл. Метод наиболее широко применяется при шении задач, мало отличающихся (в смысле ограничений) ранее решенных (например, задачи с дополнительными аничениями, параметрическое программирование; см. Гасс Саати [11] и Гасс [10]).

. Метод одновременного решения прямой двойственной задач.

При описании этого метода будем предполагать, что за-а линейного программирования задана в виде

$$\max \{p^T x \mid Ax = b, \ x \geqslant 0, \ b \geqslant 0\}. \quad (3.4.1)$$

соответствует двойственная задача (теорема 6 п. 2.5)

$$\min \{b^T u \mid A^T u - v = p, \ v \geqslant 0\}. \quad (3.4.2)$$

овие $b \geqslant 0$ не уменьшает общности, так как его всегда но соблюсти, умножая некоторые строки на -1.

Покажем, как получить возможное решение $u^{\nu+1}$ из воз-ного решения u^ν так, чтобы $b^T u^{\nu+1} < b^T u^\nu$. Процесс, м образом, начинается с исходного возможного реше-u^0 двойственной задачи (u^0 может не быть базисным нием).

Пусть $J_\nu \subset J$ определяется условием

$$J_\nu = \{j \in J \mid v_j^\nu = 0\}. \quad (3.4.3)$$

зя теперь ограниченную прямую задачу[1])

$$x \left\{ -\sum_{i=1}^m z_i \,\middle|\, \sum_{j \in J_\nu} a_{ij} x_j + z_i = b_i, \ x_j \geqslant 0, \ z_i \geqslant 0 \right\}. \quad (3.4.4)$$

лучаем оптимальное решение z_i', x_j', а также $d'(z_i)$ и $d'(x_j)$. огут встретиться два случая:

1. $$-\sum_{i=1}^m z_i' = 0.$$

[1]) Ограниченную задачу часто называют вспомогательной задачей. — *Прим. ред.*

Вектор x' удовлетворяет условиям $Ax'=b$, $x'^T v^\nu = 0$, так что, в силу теорем 5 и 6 п. 2.5, x' мальное решение (3.4.1).

2. $$-\sum_{i=1}^{m} z_i' < 0.$$

В силу (3.2.20), решение задачи, двойственной задаче имеет вид $u_i' = d'(z_i) - 1$, $v_j' = d'(x_j)$. Поэтому ремы 7 п. 2.5

$$-\sum_{i=1}^{m} z_i' = \sum_{i=1}^{m} b_i (d'(z_i) - 1) < 0.$$

Далее, при $j \in J_\nu$ $v_j = 0$ и

$$0 \leqslant d'(x_j) = v_j' = \sum_{i=1}^{m} u_i' a_{ij} = \sum_{i=1}^{m} (d'(z_i) - 1) a_{ij}.$$

Положим теперь

$$u_i = u_i^\nu + \lambda (d'(z_i) - 1)$$

и

$$\lambda_\nu = \max \left\{ \lambda \,\middle|\, \sum_{i=1}^{m} a_{ij} u_i \geqslant p_j \right\}.$$

Тогда на основании (3.4.6) $\lambda_\nu > 0$. Если $\lambda_\nu = \infty$, задача имеет неограниченное решение, поскольку выполняется т. е. задача (3.4.1) неразрешима. Положим

$$u_i^{\nu+1} = u_i^\nu + \lambda_\nu (d'(z_i) - 1).$$

Тогда $u^{\nu+1}$ — возможное решение (3.4.2) и $b^T u^{\nu+1}$ Из (3.4.8) и условия $\lambda_\nu < \infty$ следует также, что по мере для одного j $v_j^{\nu+1} = 0$ и $v_j^\nu > 0$.

Мы получили новое множество $J_{\nu+1}$ и можем пост новую ограниченную прямую задачу типа (3.4.4). В пр ложении невырожденности выполняется неравенство

$$\left\{ \sum_{i=1}^{m} z_i' \right\}^{\nu+1} < \left\{ \sum_{i=1}^{m} z_i' \right\}^{\nu}.$$

Поскольку произвольный базис ограниченной задачи является подматрицей порядка m матрицы (A, E) и каж

базису соответствует определенное значение $\sum_{i=1}^{m} z_i$, то в процессе решения базис не может встретиться дважды. Поэтому после конечного числа шагов мы получим оптимальное решение (3.4.4), значение которого $\sum_{i=1}^{m} z_i = 0$.

Метод одновременного решения прямой и двойственной задач (Данциг, Форд и Фулкерсон [15]) был впервые применен для решения весьма специальной задачи линейного программирования: так называемой задачи определения максимального потока через сеть (см. Форд и Фулкерсон [31]; Йовелл [21]), в которой для решения ограниченной задачи используется особый прием, основанный на понятии максимального потока. В задачах подобного типа всегда легко получить исходное возможное решение. В общей задаче это, как правило, невозможно, так что приходится применять искусственные приемы. К этому вопросу мы вернемся в п. 9.4, где будет показано, что метод одновременного решения прямой и двойственной задачи представляет частный случай более общего метода возможных направлений.

итерации. Поэтому справедливо соотношение

$$\bar{A}^\nu = \{A^\nu, E\},$$

где A^ν — внебазисная матрица на ν-й итерации.

Приведем формулы преобразования в развернутом

$$\bar{a}^\nu_{ij} = \bar{a}^{\nu-1}_{ij} + \bar{a}^{\nu-1}_{r_\nu j} (\eta^\nu_{r_\nu})_i, \qquad i \neq r_\nu,$$

$$\bar{a}^\nu_{r_\nu j} = \bar{a}^{\nu-1}_{r_\nu j} (\eta^\nu_{r_\nu})_{r_\nu}.$$

Отсюда следует, что столбец остается неизменным, его элемент $\bar{a}^{\nu-1}_{r_\nu j}$, принадлежащий главной строке, нулю. Если в вычислительной машине столбцы матр хранятся на магнитной ленте в сжатом виде, т. е. запис только отличные от нуля элементы и номера соответств щих строк, мы должны на ν-й итерации считывать матрицу и записывать матрицу ${}_0\bar{A}^\nu$. Преобразование каждого сто влечет, таким образом, за собой:

a) поиск элемента столбца, расположенного в гла строке;

b) перезапись столбца, если этот элемент нулевой;

c) преобразование и перезапись столбца, если этот мент отличен от нуля.

Во время преобразования столбцов последовательн сравнениями выбирается наилучшее d^ν_j (если оно существ и соответствующий главный столбец следующей итера Столбец b^ν лучше всего хранить за матрицей, так как в случае его можно сразу извлечь при переходе к пер операции.

4.3. Алгоритм с обратной матрицей.

В этом алгоритме на каждой итерации преобразу только матрица, обратная расширенной матрице базиса столбец b. Вследствие этого мы не располагаем глав столбцом $\bar{a}^{\nu-1}_{\cdot s_\nu}$, и его необходимо вычислить преобразо нием столбца $\bar{a}_{\cdot s_\nu}$ исходной матрицы. Эту операцию бу называть вычислением главного столбца.

Величины d^ν_j также приходится вновь вычислять на каждой итерации. Алгоритм может быть подразделен на следующие операции:

1) вычисляется главный столбец:

$$a^{\nu-1}_{\cdot s_\nu} = \{B^{\nu-1}\}^{-1} a_{\cdot s_\nu} \quad \text{при} \quad 1 \leqslant s_\nu \leqslant n, \qquad (4.3.1)$$

$$a^{\nu-1}_{\cdot s_\nu} = \{B^{\nu-1}\}^{-1} e_l \quad \text{при} \quad s_\nu = n + l; \qquad (4.3.2)$$

2) с помощью (3.2.15) определяется направляющий элемент главного столбца;

3) строится новый η-вектор [по формулам (3.2.16) и (3.2.17)];

4) преобразуется столбец ограничений:

$${}_0b^\nu = {}_0E^\nu_{r_\nu}\, {}_0b^{\nu-1}; \qquad (4.3.3)$$

5) преобразуется расширенная обратная матрица:

$$\{{}_0B^\nu\}^{-1} = {}_0E^\nu_{r_\nu} \{{}_0B^{\nu-1}\}^{-1}; \quad {}_0B^0 = E; \qquad (4.3.4)$$

6) выполняются операции оценки, т. е. вычисляются значения d_j и из исходной матрицы выбирается главный столбец (если он существует):

$$d^\nu_{1j} = \beta^\nu_0 a_{\cdot j}, \qquad j \notin I^\nu, \qquad (4.3.5)$$

$$d^\nu_{2i} = (\beta^\nu_0)_i, \qquad i \notin I^\nu. \qquad (4.3.6)$$

Эти формулы немедленно следуют из (3.2.3), (3.2.4) и определения β_0 и ${}_0a_{\cdot j}$.]

Операции 1, 5 и 6 называются *основными*. Операции оценки для переменных, принадлежащих текущему базису, производить не нужно, так как $d^\nu_j = 0$ при $j \in I^\nu$. Обратная матрица $\{B^\nu\}^{-1}$ содержит столько единичных столбцов, сколько дополнительных переменных остается в базисе на ν-й итерации. Если в базис вводится дополнительная переменная, то на первой операции вместо вычисления главного столбца просто берется соответствующий столбец обратной матрицы. Если в базисе содержится много дополнительных переменных, то обратная матрица содержит много единичных векторов и операции 1 и 5 относительно нетрудоемки. Операции оценки в этом случае также требуют меньших вычислений

на столбец, поскольку компоненты β_0^ν, соответству дополнительным переменным, равны нулю. Однако этом приходится вычислять большее число оценок.

На первой операции нам приходится считывать об ную матрицу, на пятой—считывать и записывать ее, на ш операции мы считываем исходную матрицу A, так обратную матрицу мы должны считывать дважды. При рочном хранении обратной матрицы можно считыва один раз, если произвести небольшое переупорядочи операций. Мы получим следующую схему:

1) предположим, что известны векторы $\eta_{r_\nu}^\nu$ и b^ν, а трица $\{B^{\nu-1}\}^{-1}$ записана на ленте построчно. Предпол далее, что известна главная строка $\beta_{r_\nu}^{\nu-1}$ обратной матр

2) считывается и преобразуется $\beta_0^{\nu-1}$:

$$\beta_0^\nu = \beta_0^{\nu-1} + \beta_{r_\nu}^{\nu-1}\left(\eta_{r_\nu}^\nu\right)_0; \qquad (4.$$

3) считывается исходная матрица и производятся оп ции оценки. Определяется $d_{s_{\nu+1}}^\nu$ и $s_{\nu+1}$, если $\bar{d}_j^\nu < 0$ некоторых j, и находится $a_{\cdot s_{\nu+1}}$, если в базис ввод основная переменная. Записывается β_0^ν;

4) считываются и преобразуются строки $\beta_i^{\nu-1}$ $(i>0)$:

$$\beta_i^\nu = \beta_i^{\nu-1} + \beta_{r_\nu}^{\nu-1}\left(\eta_{r_\nu}^\nu\right)_i, \qquad i \neq r_\nu, \qquad (4.$$

$$\beta_{r_\nu}^\nu = \beta_{r_\nu}^{\nu-1}\left(\eta_{r_\nu}^\nu\right)_{r_\nu}; \qquad (4.$$

5) вычисляются

$$\bar{a}_{is_{\nu+1}}^\nu = \beta_i^\nu a_{\cdot s_{\nu+1}} \quad \text{при} \quad 1 \leqslant s_{\nu+1} \leqslant n, \qquad (4.3$$

$$\bar{a}_{is_{\nu+1}}^\nu = \left(\beta_i^\nu\right)_l \quad \text{при} \quad s_{\nu+1} = n + l; \qquad (4.3$$

если $\bar{a}_{is_{\nu+1}}^\nu > 0$, вычисляется $\lambda_i^{\nu+1} = \dfrac{b_i^\nu}{\bar{a}_{is_{\nu+1}}^\nu}$, в то время выбирается $\lambda_{r_{\nu+1}}^{\nu+1} = \min \lambda_i^{\nu+1}$ и $\beta_{r_{\nu+1}}^\nu$. Записывается

Операции 4, 5 и 6 повторяются до тех пор, пок будут преобразованы все строки обратной матрицы;

7) строится новый η-вектор $\eta_{r_{\nu+1}}^{\nu+1}$;

8) преобразуется b^ν. Мы теперь снова подготовлены операции 1, так что итерация полностью завершена.

Заметим, что при $\left(\eta_{r_\nu}^\nu\right)_l = 0$ операция 4 не содержит никаких преобразований. Она также не содержит операций поиска. Когда строки хранятся в сжатом виде, поиск требуется лишь на операции 5, если в базис вводится дополнительная переменная [формула (4.3.11)].

В замечании в конце п. 4.5 будет показано, как можно усовершенствовать операцию 6 (операцию оценки) алгоритма обратной матрицей.

Алгоритм с обратной матрицей был предложен Чарнсом часто называется модифицированным симплексным методом. Он описан в [13] и [10 гл. 6].

4.4. Мультипликативный алгоритм.

В этом алгоритме обратная матрица $\{_0B^\nu\}^{-1}$ представляется не в явном виде, а как произведение элементарных матриц $E_{r_\rho}^\rho$, $\rho = 1, \ldots, \nu$, так что нам приходится хранить лишь η-векторы. Действительно, из (4.3.4) следует, что

$$\{_0B^\nu\}^{-1} = {}_0E_{r_\nu}^\nu \, {}_0E_{r_{\nu-1}}^{\nu-1} \cdots {}_0E_{r_1}^1. \qquad (4.4.1)$$

Ясно, что здесь мы не располагаем вектором β_0^ν, и его приходится вычислять. С другой стороны, отпадает операция преобразования обратной матрицы, имеющаяся в алгоритме с обратной матрицей. Алгоритм можно подразделить на следующие операции:

1) вычисляется главный столбец

$$\bar{a}_{\cdot s_\nu}^{\nu-1} = E_{r_{\nu-1}}^{\nu-1} E_{r_{\nu-2}}^{\nu-2} \cdots E_{r_1}^1 \bar{a}_{\cdot s_\nu}; \qquad (4.4.2)$$

2) по формулам (3.2.15) в главном столбце находится направляющий элемент;

3) строится новый η-вектор [по формулам (3.2.16) и (3.2.17)];

4) преобразуется столбец ограничений [по формуле (4.3.3)];

5) вычисляется β_0^ν:

$$\beta_0^\nu = e_0^T {}_0E_{r_\nu}^\nu {}_0E_{r_{\nu-1}}^{\nu-1} \dots {}_0E_{r_1}^1;$$

6) производятся операции оценки [по формулам (4... и (4.3.6)] и выбирается следующий главный столбец.

Операции 1, 5 и 6 основные. Операция 1 сос... в последовательном вычислении столбцов $\bar a_{\cdot s_\nu}^\rho$ $(0 \leqslant \rho \leqslant \nu ...$

$$\bar a_{\cdot s_\nu}^\rho = E_{r_\rho}^\rho a_{\cdot s_\nu}^{\rho-1}. \quad (4...$$

Следовательно, если $\bar a_{r_\rho s_\nu}^{\rho-1} = 0$, то $\bar a_{\cdot s_\nu}^\rho = \bar a_{\cdot s_\nu}^{\rho-1}$. Отсюда... что лучше всего начать с „развертывания"[1]) s_ν-го стол... тогда отпадает необходимость в поисках и переписыв... в течение всей операции 1. Мы начинаем вычисления с ...вого η-вектора и последовательно двигаемся к последне...

Определим вектор-строку $\beta_0^{\nu,\rho}$, $0 \leqslant \rho \leqslant \nu$, равенств...

$$\beta_0^{\nu,0} = e_0^T, \quad \beta_0^{\nu,\rho+1} = \beta_0^{\nu,\rho} \cdot {}_0E_{r_{\nu-\rho}}^{\nu-\rho}. \quad (4.$$

Тогда $\beta_0^{\nu,\nu} = \beta_0^\nu$ и

$$\{\beta_0^{\nu,\rho+1}\}_l = \{\beta_0^{\nu,\rho}\}_l \quad \text{при} \quad l \neq r_{\nu-\rho}, \quad (4.$$

$$\{\beta_0^{\nu,\rho+1}\}_{r_{\nu-\rho}} = \sum_{l=0}^{m} \{\beta_0^{\nu,\rho}\}_l \{{}_0\eta_{r_{\nu-\rho}}^{\nu-\rho}\}_l. \quad (4.$$

Отсюда видно, что на каждом шаге операции 5 приход... вычислять только одну величину. Если в нашем распоря...нии несжатый вектор $\beta_0^{\nu,\rho}$ (т. е. мы начинаем с несжат... единичного вектора), то найденная величина может б... сразу помещена в соответствующую позицию этого векто... В начале этой операции используется последний η-вект... в конце ее — первый.

Третья основная операция, операция оценки, ана...гична соответствующей операции алгоритма с обрат... матрицей. Поведение дополнительных переменных игр... в мультипликативном алгоритме столь же важную роль, ... и в предыдущем. Чем больше дополнительных перемен... входит в базис, тем меньше вычислений приходится де...

[1]) Если он хранится в сжатом виде. — *Прим. перев.*

...перации 5. Преимущества возникают и в операции 1. ...к, если $\bar a_{\cdot s_\nu}$ — единичный вектор, например e_l, то в пер... операции $\bar a_{\cdot s_\nu}^\rho = e_l$, т. е. не требуется никаких вычис...ний до тех пор, пока ρ меньше номера итерации, на ко...ой номер главной строки равен l[1]). Аналогичные пре...ущества имеют место, когда некоторая переменная входит ...базис на ν_1-й итерации, покидает его на ν_2-й итерации, ...а ν-й ($\nu > \nu_2$) снова вводится в базис. В этом случае для ...$\leqslant \rho \leqslant \nu_2$ $\bar a_{\cdot s_\nu}^\rho$ — единичный вектор (практически из-за оши...к округления это условие может несколько нарушаться, ...да вычислительные преимущества утрачиваются).

В процессе вычислений мы получаем все больше η-век...ров, так что итерации занимают все больше и больше ...емени. Обычно требуется довольно большое число изме...ний базиса, поэтому в конце концов делается выгодным ...нать решать задачу снова с помощью так называемого *...вторения.*

Мы начинаем решение снова, но выбираем главные ...олбцы и строки так, чтобы переменные, входившие в по...едний перед повторением базис, были снова введены ...него за наименьшее возможное число итераций. Итерации ... время повторения весьма просты, так как отпадает на...бность в двух операциях (операции 5 и 6) из трех ос...вных. Столбцы, которые надо ввести в базис, выбираются ... очереди. После выбора столбца производится его пре...разование (операция „вычисление главного столбца"). На...авляющий элемент всегда выбирается из числа элементов ...олбца, расположенных в свободных строках. (Выбирается, ...пример, абсолютно наибольший элемент из расположен...х в свободных строках, т. е. таких, которые должны ...ть выбраны, но еще не были выбраны в качестве на...авляющих.)

Столбец b также последовательно преобразуется в про...ссе повторения. Таким путем мы получим значительно ...ньшее число η-векторов, т. е. последующие итерации

[1]) Если мы начинаем с единичного базиса, этот случай охва...вается следующим. — *Прим. перев.*

потребуют меньше времени. Ясно, что повторение полез с точки зрения точности вычислений.

Мультипликативный алгоритм описан, например, в и [26].

4.5. Модифицированный мультипликативный алгорит

Вычисление d-строки, необходимое для выбора сле щего главного столбца, занимает в мультипликативном горитме очень много времени, так как оно включает из трех основных операций. Внебазисные матрицы A^ν з линейного программирования больших размеров, как прав содержат большое число нулевых элементов. Это верно только в начале, но и в конце вычислений. Число от ных от нуля элементов матрицы A^ν составляет об 30—50% от общего числа mn элементов. Величина $\bar d_j$ данной итерации изменяется только тогда, когда элем j-го столбца, лежащий в направляющей строке, отличен нуля. Поэтому многие значения внебазисных $\bar d_j$, в сред 75%, не изменяются во время итерации и их вычисле излишне. d-строка может также преобразовываться на каж итерации, как это делается в прямом алгоритме. Для эт необходимо знать $\bar d^{\nu-1}$ и $\bar a^\nu_{r_\nu}$. Строка $\bar d^{\nu-1}$ может хранит с предыдущей итерации, но $\bar a^\nu_{r_\nu}$. на ν-й итерации приде вычислять. Вычисление главной строки производится так как и вычисление d-строки, т. е. состоит из двух опе ций: вычисления $\beta^\nu_{r_\nu}$ и вычисления $a^\nu_{r_\nu j} = \beta^\nu_{r_\nu} a_{\cdot j}$ для всех Но в то время как все компоненты β^ν_0, соответствующие новным переменным (т. е. строкам, где переменные x_j менили в базисе переменные y_i), по-видимому, отличны нуля, соответствующая часть строки $\beta^\nu_{r_\nu}$ в среднем так полна, как и вся внебазисная матрица. Полнота определяе как отношение между числом отличных от нуля элемен и общим числом элементов. Таким образом, в модифициров ном алгоритме на каждой итерации вычисляется главная стр и с ее помощью преобразуется d-строка. Кроме этого осн ного отличия, модифицированный алгоритм содержит и

которые другие изменения. Они ведут к существенному сокращению числа арифметических операций, а также уменьшают ожидаемое число обращений к магнитной ленте.

Дадим теперь обзор всех особенностей алгоритма.

1. На ν-й итерации вычисляется вектор $\gamma^\nu_{r_\nu}$, равный произведению $-\bar d^{\nu-1}_{s_\nu}$ на главную строку обратной матрицы:

$$\gamma^\nu_{r_\nu} = -\bar d^{\nu-1}_{s_\nu}\beta^\nu_{r_\nu} = -\bar d^{\nu-1}_{s_\nu} e^T_{r_\nu} E^\nu_{r_\nu} E^{\nu-1}_{r_{\nu-1}} \dots E^1_{r_1} \qquad (4.5.1)$$

(цель умножения на скаляр $-\bar d^{\nu-1}_{s_\nu}$ ясна из нижеследующего). Определим $\gamma^{\nu,\rho}_{r_\nu}$ для $0 \leqslant \rho \leqslant \nu$ равенствами

$$\gamma^{\nu,0}_{r_\nu} = -\bar d^{\nu-1}_{s_\nu} e^T_{r_\nu}, \quad \gamma^{\nu,\rho+1}_{r_\nu} = \gamma^{\nu,\rho}_{r_\nu} E^{\nu-\rho}_{r_{\nu-\rho}}. \qquad (4.5.2)$$

Тогда $\gamma^{\nu,\nu}_{r_\nu} = \gamma^\nu_{r_\nu}$ и

$$\{\gamma^{\nu,\rho+1}_{r_\nu}\}_l = \{\gamma^{\nu,\rho}_{r_\nu}\}_l, \quad \text{если} \quad l \neq r_{\nu-\rho}, \qquad (4.5.3)$$

$$\{\gamma^{\nu,\rho+1}_{r_\nu}\}_{r_{\nu-\rho}} = \sum_{l=1}^{m} \{\gamma^{\nu,\rho}_{r_\nu}\}_l \{\eta^{\nu-\rho}_{r_{\nu-\rho}}\}_l. \qquad (4.5.4)$$

Отсюда видно, что на каждом шаге этой операции необходимо вычислять лишь одну новую величину. Видно также, что мы можем оперировать с матрицами $E^\nu_{r_\nu}$ вместо матриц ${}_0E^\nu_{r_\nu}$, так что η-векторы содержат на одну компоненту меньше.

2. Величина $\bar d_j$ вычисляется теперь по следующим формулам (операция оценки и γ-операция):

$$d^\nu_{2i} = d^{\nu-1}_{2i} - \bar d^{\nu-1}_{s_\nu}\{\beta^\nu_{r_\nu}\}_i = d^{\nu-1}_{2i} + \{\gamma^\nu_{r_\nu}\}_i, \qquad (4.5.5)$$

$$d^\nu_{1j} = d^{\nu-1}_{1j} - \bar d^{\nu-1}_{s_\nu}\{\beta^\nu_{r_\nu} a_{\cdot j}\} = d^{\nu-1}_{1j} + \{\gamma^\nu_{r_\nu} a_{\cdot j}\}. \qquad (4.5.6)$$

Из этих формул ясно, почему лучше вычислять $\gamma^\nu_{r_\nu}$, а не $\beta^\nu_{r_\nu}$.

3. Дадим следующее определение.

Столбец (переменная) называется *неиспользованным*, если он не был базисным ни на одной итерации, начиная с последнего повторения. В противном случае столбец называется *использованным*. Предположим, что на $(\nu-\rho)$-й итерации $(1 \leqslant \rho \leqslant \nu)$ переменная $\bar x_{j_{\nu-\rho}}$ исключается из базиса, так что в $(\nu-\rho-1)$-й таблице эта переменная входит в базис и

расположена на $r_{\nu-\rho}$-м месте. При этом $\bar{a}_{\cdot j_{\nu-\rho}}^{\nu-\rho-1} = e_{r_\nu}$
$\bar{a}_{\cdot j_{\nu-\rho}}^{\nu-\rho} = \eta_{r_{\nu-\rho}}^{\nu-\rho}$. Отсюда следует, что $\{\beta_{r_\nu}^{\nu,\rho+1}\}_{r_{\nu-\rho}} = \beta_{r_\nu}^{\nu,\rho+1} e_{r_\nu}$
$= \beta_{r_\nu}^{\nu,\rho+1} \bar{a}_{\cdot j_{\nu-\rho}}^{\nu-\rho-1} = \beta_{r_\nu}^{\nu,\rho+1} E_{r_{\nu-\rho-1}}^{\nu-\rho-1} \dots E_{r_1}^{1} \bar{a}_{\cdot j_{\nu-\rho}} = \beta_{r_\nu}^{\nu} \bar{a}_{\cdot j_{\nu-\rho}}$
$= \bar{a}_{r_\nu j_{\nu-\rho}}^{\nu}$. Тогда каждое новое значение $\{\gamma_{r_\nu}^{\nu,\rho+1}\}_{r_{\nu-\rho}}$,
$\leqslant \rho \leqslant \nu - 1$, равно $\gamma_{r_\nu}^{\nu} \bar{a}_{\cdot j}$ (произведению $-\bar{d}_{s_\nu}^{\nu-1}$ на элем
преобразованной направляющей строки на ν-й итераци
величина $\bar{d}_j^{\nu}$ для использованных переменных $\bar{x}_{j_1}, \dots,$
может быть определена по формуле

$$\bar{d}_{j_{\nu-\rho}}^{\nu} = \bar{d}_{j_{\nu-\rho}}^{\nu-1} + \{\gamma_{r_\nu}^{\nu,\rho+1}\}_{r_{\nu-\rho}}, \quad \rho = 0, 1, \dots, \nu - 1, \quad (4.$$

во время γ-операции.

Если до ν-й итерации переменная дважды покидала
зис, можно просто и надежно предотвратить повторное
образование величины $\bar{d}_j$.

4. Для многих значений ρ вообще нет нужды вычисл
новые значения $\{\gamma_{r_\nu}^{\nu,\rho+1}\}_{r_{\nu-\rho}}$:

a. Если $\bar{x}_{j_{\nu-\rho}}$ — базисная переменная на ν-й итера
то $\{\gamma_{r_\nu}^{\nu,\rho+1}\}_{r_{\nu-\rho}} = -d_{s_\nu}^{\nu-1}$, если $\bar{x}_{j_{\nu-\rho}} = \bar{x}_{s_\nu}$ (т. е. если $\bar{x}_j$
находится на r_ν-м месте в ν-м базисе), и $\{\gamma_{r_\nu}^{\nu,\rho+1}\}_{r_{\nu-\rho}} =$
в противном случае.

b. Если $\bar{x}_{j_{\nu-\rho}}$ исключалась из базиса на некоторой ит
рации между (ν — ρ)-й и ν-й, то величина $\{\gamma_{r_\nu}^{\nu,\rho+1}\}_{r_{\nu-\rho}}$ вычи
лялась нами, и не нужно вычислять ее снова при услови
что мы запоминаем такие величины в тех случаях, ког
они могут понадобиться вновь. Конечно, это требует д
полнительной памяти, но может быть сделано без чрезме
ных усложнений.

5. Ясно, что в процессе вычислений все большее числ
столбцов становится использованными и соответствующ
значения $\bar{d}_j^{\nu}$ можно находить при вычислении $\gamma_{r_\nu}^{\nu}$. Поэтом
в операциях оценки внебазисные столбцы исходной матриц
фигурируют до тех пор, пока они не станут использован
ными. Обычно исходная матрица хранится на ленте. Поэтому

разумно периодически очищать ленту от использованных столбцов. Это сокращает время обращения к ленте в течение операции оценки.

6. Если использованная переменная, исключенная из базиса на (ν — ρ)-й итерации, снова входит в базис на ν-й итерации, то

$$\bar{a}_{\cdot s_\nu}^{\nu-1} = E_{r_{\nu-1}}^{\nu-1} \dots E_{r_{\nu-\rho+1}}^{\nu-\rho+1} \bar{a}_{\cdot s_\nu}^{\nu-\rho} = E_{r_{\nu-1}}^{\nu-1} \dots E_{r_{\nu-\rho+1}}^{\nu-\rho+1} \eta_{r_{\nu-\rho}}^{\nu-\rho}, \quad (4.5.8)$$

т. е. вычисление главного столбца может начинаться с (ν—ρ)-го η-вектора. Число ν — ρ можно просто получить при вычислении γ, так как при последовательном расчете по формулам (4.5.7) величин $\bar{d}_j$, соответствующих использованным столбцам, мы можем запоминать номер ν — ρ, когда полученное значение $\bar{d}_{j_{\nu-\rho}}^{\nu}$ меньше всех ранее вычисленных значений. В конце операций вычисления γ и оценки мы, таким образом, будем знать, какой столбец вводится в базис, использованный или неиспользованный. Если в базис вводится использованный столбец, мы будем располагать номером ν — ρ соответствующей итерации.

7. При повторении известно, какие столбцы исходной матрицы надо выбрать в качестве главных. Вместо того чтобы выбирать их поочередно, выгоднее сначала выбрать столбец, имеющий при данном базисе меньшее число отличных от нуля элементов. В этом случае η-векторы будут, вообще говоря, обладать меньшей полнотой. Но для того чтобы сравнивать столбцы на ρ-й итерации повторения, все они должны быть преобразованы, так как мы должны сравнивать столбцы $a_{\cdot j}^{\rho}$, а не $a_{\cdot j}$. Мы предлагаем поэтому на каждой итерации повторения последовательно преобразовывать все столбцы, которые входили в последний перед повторением базис и не выбирались еще в качестве главных. Критерием выбора главного столбца будет наименьшее число ненулевых элементов в столбце. (В случае неединственности выбирается первый из столбцов.) Направляющий элемент в главном столбце выбирается так, чтобы он a) принадлежал свободной строке, b) был не слишком малым, c) принадлежал строке, обладающей наименьшим среди оставшихся строк числом ненулевых элементов.

Каждый новый преобразованный главный столбец являет очередным η-вектором. Он больше не преобразуется п повторении. Можно ожидать, что при таком способе выбо направляющего столбца и направляющей строки полно η-векторов будет существенно меньше, т. е. уменьши число арифметических операций и операций обращения ленте в процессе повторения. Этот метод представляется в краней мере столь же хорошим, как и метод исключени Марковица [25], но менее сложным. Столбец в процессе п вторения преобразуется последовательно. После повторения н обходимо вновь вычислить d-строку и перегруппировать стол цы на магнитной ленте, выделяя неиспользованные столбц

Если известен хороший исходный базис, можно начина вычисления с операций повторения, называемых в этом сл чае *предварительными*. Во время предварительных опер ций можно не выбирать главными столбцы, в которых н достаточно больших направляющих элементов среди элеме тов свободных строк. Предварительные операции полезн в частности, когда определяется решение задачи с матриц мало отличающейся от матрицы ранее решенной задачи. О тимальный базис решенной задачи можно выбрать как и ходный в новой задаче. Если выбранная матрица оказывает особенной, то нужно лишь на предварительных операци один или несколько главных столбцов не выбирать главным Однако при этом можно получить столбец b с отрицател ными компонентами, поэтому мы должны располагать эффе тивным способом действий в подобной ситуации.

Для специальной задачи такой способ будет опис в конце п. 6.3. Из изложенного выше ясно, что модиф цированный мультипликативный алгоритм состоит из следую щих операций:

1) вычисление главного столбца по формуле (4.4.2), ес в базис вводится неиспользованный столбец, и по фор муле (4.5.8), если столбец использованный;

2) отыскание направляющего элемента в главном столб по формулам (3.2.15);

3) построение нового η-вектора по формулам (3.2.16 и (3.2.17);

4) преобразование столбца ограничений по формула (4.3.3);

5) вычисление γ по формулам (4.5.3) и (4.5.4).

Для преобразования величин $\bar{d}_j$, соответствующих использованным переменным, выбора среди них наибольшего по абсолютной величине отрицательного значения (если оно существует), а также для выбора номера итерации соответствующего η-вектора используются формулы (4.5.7);

6) выполнение оценки по формулам (4.5.6) для неиспользованных столбцов исходной матрицы. Замена выбранного на операции 5 значения $\bar{d}_j$ еще меньшим (если оно существует) и выбор соответствующего столбца в качестве главного.

Операции 1, 5, 6 являются снова основными.

В модифицированном алгоритме, как и в других алгоритмах, вычисления упрощаются, когда в базисе много дополнительных переменных. Но здесь лучше используется поведение переменных, то входящих в базис, то покидающих его. Такие переменные требуют меньшего счета в операциях 1 и 5. Алгоритм был разработан прежде всего с целью максимального сокращения числа арифметических операций, что представляется наиболее важным в настоящее время, когда вычислительные машины обладают огромной внешней памятью, а арифметические операции и операции обращения к ленте выполняются одновременно. Однако в гл. 5 показано, что модифицированный алгоритм требует также меньшего числа обращений к внешней памяти, чем любой другой алгоритм.

З а м е ч а н и е. Ясно, что число арифметических операций в алгоритме с обратной матрицей также может быть значительно сокращено, если вместо строки β_0^{ν} мы будем использовать $\beta_{r_\nu}^{\nu}$ и преобразовывать строку d на каждой итерации по формулам (4.5.5) и (4.5.6).

Нетрудно сделать соответствующие изменения в вычислительной схеме, приведенной в конце п. 4.3.

4.6. Вычислительные алгоритмы двойственного симплексного метода.

Для двойственного симплексного метода можно также разработать различные вычислительные алгоритмы.

Будем предполагать, что столбцы хранятся в сжатом виде.

a. *Прямой алгоритм.* Очередной номер направляюще строки определяется при преобразовании столбца b. Это столбец в памяти должен теперь предшествовать матриц

Величины (3.3.1) вычисляются во время преобразован столбцов расширенной матрицы. Предполагаемые главны столбцы могут храниться в оперативной памяти. Тогда посл преобразования можно сразу делить элементы главно столбца на величину направляющего элемента, затем преоб разованием b начинать следующую итерацию и т. д.

b. *Алгоритм с обратной матрицей.* Приведем схем вычислений, при которой обратная матрица считываетс единственный раз на каждой итерации. Предположим, чт строки $\beta_i^{\nu-1}$ обратной матрицы записаны на ленте, строка $\beta_{r_\nu}^{\nu-1}$ в накопителе (r_ν — номер главной строки на следующе итерации), векторы $b^{\nu-1}$ и $\bar d^{\nu-1} = \begin{pmatrix} d_1^{\nu-1} \\ \beta_0^{\nu-1} \end{pmatrix}$ вычислены. Буде действовать по следующей схеме:

1) вычислим отношения $\{\beta_0^{\nu-1}\}_i / - \{\beta_{r_\nu}^{\nu-1}\}_i$ для тех i, для которых знаменатель положителен, выберем наименьше из них и соответствующий номер столбца s_ν;

2) вычислим вектор-строку $\beta_{r_\nu}^{\nu-1}A$ и отношени $d_{1j}^{\nu-1}/-\beta_{r_\nu}^{\nu-1}a_{\cdot j}$ для тех j, для которых знаменатель положителен

Если полученное отношение меньше вычисленных ране на операциях 1 и 2, то соответствующий столбец $a_{\cdot j}$ записывается в оперативную память;

3) если положительных знаменателей нет, то задача неразрешима. В противном случае переходим к операции 4;

4) если $0 < s_\nu \leqslant n$, вычислим $\{\eta_{r_\nu}^\nu\}_{r_\nu} = \dfrac{1}{a_{r_\nu s_\nu}^{\nu-1}}\left(a_{r_\nu s_\nu}^{\nu-1} - s_\nu\right.$ элемент строки $\left.\beta_{r_\nu}^{\nu-1}.A\right)$.

Если $s_\nu = n + l$, найдем $\{\beta_{r_\nu}^{\nu-1}\}_l$ и вычислим

$$\{\eta_{r_\nu}^\nu\}_{r_\nu} = \frac{1}{\{\beta_{r_\nu}^{\nu-1}\}_l};$$

5) преобразуем $\beta_0^{\nu-1}$ и $d_1^{\nu-1}$:

$$\beta_0^\nu = \beta_0^{\nu-1} + \beta_{r_\nu}^{\nu-1}\{\eta_{r_\nu}^\nu\}_0,$$

$$d_1^\nu = d_1^{\nu-1} + (\beta_{r_\nu}^{\nu-1}.A)\{\eta_{r_\nu}^\nu\}_0,$$

где $\{\eta_{r_\nu}^\nu\}_0 = \bar d_{s_\nu}^{\nu-1} / - \bar a_{r_\nu s_\nu}^{\nu-1}$, что немедленно следует из операции 1 или 2.

6) считываем строки $\beta_l^{\nu-1}$ обратной матрицы $(1 \leqslant l \leqslant m)$, вычисляем $\bar a_{ls_\nu}^{\nu-1} = \beta_l^{\nu-1} a_{\cdot s_\nu}$ (или $\bar a_{ln+l}^{\nu-1} = \{\beta_l^{\nu-1}\}_l$), вычисляем $\{\eta_{r_\nu}^\nu\}_l = -\bar a_{ls_\nu}^{\nu-1}\{\eta_{r_\nu}^\nu\}_{r_\nu}$ и преобразуем $\beta_l^{\nu-1}$:

$$\beta_l^\nu = \beta_l^{\nu-1} + \{\eta_{r_\nu}^\nu\}_l \beta_{r_\nu}^{\nu-1};$$

вычисляем $b_l^\nu = b_l^{\nu-1} + \{\eta_{r_\nu}^\nu\}_l b_{r_\nu}^{\nu-1}$. Если $b_l^\nu < 0$ и меньше любой вычисленной ранее компоненты b^ν, записываем β_l^ν и l в оперативную память в качестве возможной главной строки обратной матрицы и ее номера соответственно, в противном случае записываем β_l^ν на ленту.

7) когда операция 6 выполнена для всех l, можно вернуться к операции 1.

c. *Модифицированный мультипликативный алгоритм.* Дадим схему этого алгоритма:

1) вычислим $\beta_{r_\nu}^{\nu-1}$, элементы $\beta_{r_\nu}^{\nu-1}\bar a_{\cdot j}$ и частные (3.3.1) для использованных переменных. Перепишем наименьшее частное и соответствующий η-вектор в оперативную память;

2) вычислим $\beta_{r_\nu}^{\nu-1}a_{\cdot j}$ для всех неиспользованных столбцов, вычислим частные (3.3.1). Если какое-либо частное меньше всех вычисленных ранее, запишем соответствующий столбец в оперативную память;

3) если на операциях 1 и 2 не было вычислено ни одного частного, то задача неразрешима. В противном случае переходим к операции 4;

4) эта операция проводится аналогично операции 5 двойственного алгоритма с обратной матрицей;

5) вычислим главный столбец по формулам (4.4.2) или (4.5.8);

6) строим новый η-вектор;

7) преобразуем вектор ограничений и выбираем номер следующей направляющей строки.

Повторение может производиться периодически, как описано в п. 4.5.

Глава 5

ЧИСЛЕННОЕ СРАВНЕНИЕ РАЗЛИЧНЫХ АЛГОРИТМОВ

5.1. Введение.

В этой главе делается попытка сравнить различные в числительные алгоритмы симплексного метода при испо зовании быстродействующих электронных вычислитель машин. Основное внимание уделяется задачам линейн программирования больших размеров, содержащим, скаж 300 или более ограничений и 500 или более внебазис переменных. Предположим, что матрица содержит мало э ментов, отличных от нуля (например, 2—3%), и это и пользуется программой счета, так что умножение или сл жение производятся лишь тогда, когда оба сомножителя и слагаемых отличны от нуля. В п. 5.2 кратко рассмотре различные критерии сравнения алгоритмов. В п. 5.3 о суждается поведение матриц задачи линейного программир вания, когда последняя решается симплексным методо В п. 5.4 описаны принятые нами допущения, необходим для получения сравнительных результатов, приведенн в п. 5.5.

В этой главе используются следующие сокращени обозначения:

ПА — прямой алгоритм;

ОМА — алгоритм с обратной матрицей;

МА — мультипликативный алгоритм;

ММА — модифицированный мультипликативный алгори

5.2. Критерии сравнения.

До появления вычислительных машин с ускоренн обращением к внешней памяти и совмещением арифмет ческих операций и операций пересылки между лентой и б

рабаном лучшим был алгоритм, требующий наименьшего числа пересылок.

Действительно, несколько лет назад это послужило основным стимулом для разработки мультипликативного алгоритма (см. Орчард-Хейс [26]).

Для современных вычислительных машин этот аргумент утратил свое значение, и сравнение стало более сложным, так как приходится принимать во внимание многие другие факторы. Назовем некоторые из них:

a. Число арифметических операций.

Во всех алгоритмах умножение, как правило, сопровождается сложением, а число делений относительно мало (деление производится лишь на двух неосновных операциях, к тому же совпадающих во всех алгоритмах). Поэтому полное время, необходимое на арифметические операции, приблизительно пропорционально числу умножений.

b. Число вспомогательных операций.

Некоторые из подобных операций совпадают для всех алгоритмов, например изменения номеров базисных векторов на каждой итерации. Число других операций прямо пропорционально числу умножений. Например, в операции $p = a + bc$, характерной для симплексного метода, всегда перед сложением проверяется „$a = 0$?“. Также проверяется „$|p| \ll |a|$?“. (Если p не равно нулю, но существенно меньше $|a|$, то предполагается, что p отличается от нуля лишь благодаря ошибкам округления.) Некоторые вспомогательные операции зависят от построения программы. При использовании ПА и постолбцовом сжатом хранении возникает много операций поиска и перезаписи (см. п. 4.2). Едва ли подобные операции возникают в ОМА. В МА и ММА они наверняка не возникают.

c. Число операций пересылки между лентой и барабаном.

Это число зависит, естественно, от применяемой вычислительной машины, от объема ее оперативной памяти и от количества обрабатываемой информации. Поэтому в качестве критерия примем количество чисел, с которыми мы оперируем. Если быстродействующая память не используется для постоянного хранения исходной информации (матрица, обратная матрица или η-векторы), число пересылок приблизительно пропорционально количеству этих чисел.

d. Точность.

e. Гибкость программы; например, возможность возоб[illegible] вить вычисления.

f. Сложность программы.

5.3. Поведение практических задач линейного программирования.

Чтобы иметь возможность сравнивать, необходимо сд[illegible] лать некоторые допущения о поведении задач линейного пр[illegible] граммирования больших размеров при их решении симплексн[illegible] методом. Для этого изучалось поведение ряда таких зада[illegible]

Из результатов гл. 4 следует, что наиболее важны[illegible] факторами, определяющими полное время решения, помим[illegible] алгоритма и быстродействия вычислительной машин[illegible] являются:

1) размеры задачи;

2) число итераций;

3) исходная полнота матрицы и рост полноты в про[illegible] цессе решения (полнота матрицы определяется отношени[illegible] числа ненулевых элементов внебазисной матрицы к числ[illegible] всех элементов mn);

4) поведение дополнительных переменных, особен[illegible] число переменных, исключаемых из базиса (важно тольк[illegible] в ОМА, МА и ММА);

5) число повторений в МА и ММА и их периодичност[illegible]

6) выполнение арифметических операций с обычным и[illegible] с удвоенным числом разрядов.

Так как для частной задачи размеры фиксирован[illegible] а число итераций одинаково для всех алгоритмов (обычн[illegible] лежит в пределах от m до $2,5\,m$), мы рассмотрим лиш[illegible] последние четыре фактора.

Изучение различных задач больших размеров показы[illegible] вает, что в большинстве случаев полнота довольно быстр[illegible] растет на первых итерациях (в течение примерно одной[illegible] трети их общего числа), после чего колеблется около по[illegible] стоянного уровня, в некоторых случаях на последних ите[illegible] рациях даже падает. На первых итерацях дополнительные базисные переменные почти всегда заменяются основными. За этими итерациями следует значительное число итераций, на которых число дополнительных базисных переменны[illegible] уменьшается незначительно. Многие дополнительные пере-

[illegible]менные (до 30–40% от общего числа) входят в оптималь[illegible]ный базис. Что касается повторений, то опыт показывает, что время решения сильно зависит от их числа. Однако не[illegible]существенно, начинается ли повторение несколькими итера[illegible]циями раньше или позже. Производить повторение в первой части вычислений, когда дополнительные переменные поки[illegible]дают базис и все еще быстро растет полнота, не имеет смысла. Повторения должны производиться периодически. Опыт показывает, что для задачи линейного программиро[illegible]вания весьма больших размеров получаемая точность удо[illegible]влетворительна при 25—30 двоичных разрядах в машинах с плавающей запятой. Этот факт, вероятно, объясняется тем, что в практических задачах больших размеров полнота матриц весьма мала и они имеют специфическое строение.

Удвоенная разрядность желательна лишь при повторе[illegible]ниях в задачах с числом ограничений, превосходящим 400–500. На этом опыте основываются утверждения, сделанные в следующем пункте, необходимые для теоретического сравнения алгоритмов.

5.4. Предположения о поведении задач линейного программирования.

Обозначения:

m — число строк матрицы;

n — число столбцов;

k — общее число итераций;

$\varphi(\nu)$ — полнота внебазисной матрицы на ν-й итерации,

$$\varphi(0) = \pi; \quad \varphi(k) = \varphi;$$

$\tau = \dfrac{\varphi - \pi}{\alpha m}$ — отношение приращения полноты к некоторому числу итераций ($\alpha > 0$);

$\sigma(\nu)$ — число дополнительных векторов в базисе на ν-й итерации.

Предположения:

1. О дополнительных векторах.

a. В ОМА:

$$\sigma(\nu) = \begin{cases} \nu & \text{при } 1 \leqslant \nu \leqslant \alpha m, \; 0 < \alpha \leqslant 1, \\ \alpha m & \text{при } \nu \geqslant \alpha m; \end{cases}$$

в МА и ММА:

$$\sigma(\nu)=\begin{cases}\nu & \text{при } 1\leqslant\nu\leqslant\alpha_0 m,\quad 0<\alpha_0\leqslant 1\\ \alpha_0 m & \text{при } \nu\geqslant\alpha_0 m, \text{ повторения не производилось;}\\ \alpha_l m & \text{на } l\text{-й стадии, т. е. после } l\text{-го повторения.}\end{cases}$$

b. В ОМА:

При $\nu \leqslant \alpha m$ ни один дополнительный вектор не входит в базис вновь; при $\nu > \alpha m$ с вероятностью γ' на каждой итерации некоторый дополнительный вектор вновь входит в базис;

в МА и ММА:

При $\nu \leqslant \alpha_0 m$ ни один дополнительный вектор не входит в базис вновь, при $\nu > \alpha_0 m$ и после проведения l повторений на каждой итерации с вероятностью γ'_l некоторый дополнительный вектор вновь входит в базис.

2. О полноте.

a. В ПА и ОМА полнота внебазисной матрицы, части обратной матрицы, соответствующей основным векторам и каждого столбца матрицы растет линейно на первых итерациях ($\beta \geqslant \alpha$ в ОМА), после чего остается постоянной:

$$\varphi(\nu)=\begin{cases}\pi+\nu\tau & \text{при } 1\leqslant\nu\leqslant\beta m,\\ \varphi & \text{при } \nu\geqslant\beta m,\end{cases}$$

где $\tau = \frac{\varphi-\pi}{\beta m}$.

В МА и ММА до первого повторения (стадия 0)

$$\varphi(\nu)=\begin{cases}\pi+\nu\tau_0 & \text{при } 1\leqslant\nu\leqslant\alpha_0 m,\\ \varphi_0 & \text{при } \nu\geqslant\alpha_0 m,\end{cases}$$

где $\tau_0 = \frac{\varphi_0-\pi}{\alpha_0 m}$; после l-го повторения (на l-й стадии, $l >$ 0)

$$\varphi(\nu)=\varphi_l \text{ (постоянно).}$$

Во время повторения в МА

$\varphi(\nu_l)=\pi+\tau_l\nu_l$ (ν_l — номер итерации во время повторения),

где $\tau_l = \frac{\varphi_l-\pi}{\alpha_l m}$ ($\alpha_l m$ — число итераций в l-м повторении),

Во время повторения в ММА

$$\varphi(\nu_l)=\pi+\frac{\tau_l}{\alpha_l m}\nu_l^2.$$

Здесь мы предполагаем квадратичную зависимость полноты от номера итерации.

Заметим, что здесь также $\varphi(\alpha_l m)=\varphi_l$.

b. При полноте $\varphi(\nu)$ каждый элемент матрицы с одинаковой вероятностью $\varphi(\nu)$ отличен от нуля.

Вероятность того, что произведение bc отлично от нуля, равна $\{\varphi(\nu)\}^2$, если неизвестно заранее, что b или c отлично от нуля.

c. Главные столбцы и часть направляющих строк, соответствующая основным столбцам, в среднем обладают той же полнотой, что и внебазисная матрица, т. е. $\varphi(\nu)$.

3. Число элементов $\beta_0^{\nu,\rho}$ и $\gamma_{r_\nu}^{\nu,\rho}$, отличных от нуля,

равно ρ в $\beta_0^{\nu,\rho}$ и $\rho\varphi(\nu)$ в $\gamma_{r_\nu}^{\nu,\rho}$ при $\rho\leqslant\alpha_l m$,

равно $\alpha_l m$ в $\beta_0^{\nu,\rho}$ и $\alpha_l m\varphi(\nu)$ в $\gamma_{r_\nu}^{\nu,\rho}$ при $\rho\geqslant\alpha_l m$.

4. О свойствах использованных переменных, вновь вводимых в базис.

a. На l-й стадии с вероятностью γ_l на каждой итерации в базис вводится использованный вектор (при $l=0$ эта вероятность отлична от нуля лишь для $\nu>\alpha_0 m$).

b. С вероятностью $\gamma'_l<\gamma_l$ этот использованный вектор является дополнительным и впервые вновь вводится в базис. Мы предполагаем, что этот вектор покинул базис на $\alpha_l m$-й итерации (следовательно, в операции вычисления главного столбца умножения требуются лишь при $\rho\geqslant\alpha_l m$, в ММА мы просто начинаем с $\rho=\alpha_l m$).

c. С вероятностью $\gamma''_l=\gamma_l-\gamma'_l$ этот вектор вводится в базис вторично. В этом случае мы предполагаем, что вектор введен в базис на $\alpha_l m$-й итерации и остается там в течение одной трети итераций до следующего повторения.

5. Что касается повторений, то мы принимаем:

a) что требования точности не влияют на периодичность повторений в МА и в ММА;

b) что первое повторение производится после по крайней мере $\alpha_0 m$ итераций.

6. О точности.

Либо все вычисления могут быть сделаны с обычн точностью, либо необходима удвоенная точность при повт рениях. В последнем случае принимается, что одно умн жение с удвоенной точностью эквивалентно трем с обычно Предполагаем, что для увеличения точности в ОМА так проводятся повторения с удвоенным числом разрядов.

7. Дополнительные предположения о ММА.

a. Пренебрегаем тем, что в некоторых случаях не тр буется вовсе вычислять величину $\{\gamma_{r_\nu}^{\nu,\rho}\}_{r_{\nu-\rho}}$ (см. п. 4.5; 4

b. Перезапись неиспользованных столбцов на магнитн ленте производится только после повторения.

8.a. Предполагается, что столбцы хранятся в сжатом вид т. е. хранятся только ненулевые элементы совместно с с ответствующими номерами строк. Это касается матриц А в ПА, $\{B^\nu\}^{-1}$ и A в ОМА и η-векторов и A в МА и ММ

b. В оперативной памяти нет постоянных ячеек для хр нения данных, упомянутых в п. 8а, — они хранятся либо н магнитной ленте, либо на магнитном барабане.

9. Мы будем сравнивать трудоемкость только основны операций, так как остальные занимают незначительную час времени вычислений. Более того, они одинаковы или почт одинаковы во всех алгоритмах. Таким образом, во все алгоритмах мы пренебрегаем операциями выбора направляю щего элемента, построения нового η-вектора, преобразова ния столбца b. Кроме того, мы пренебрегаем

в ПА преобразованием строки d,

в ОМА преобразованием $\beta_0^{\nu-1}$,

в МА преобразованием элементов $\{\eta_{r_\nu}^\nu\}_0$, т. е. элементо η-вектора, соответствующих d-строке,

в ММА вычислением d-строки после повторения.

Сделанные допущения, быть может, слишком произвольн и даже неверны в некоторых случаях. Однако при этом либо их влияние незначительно (например, 4b и 4c), либо время решения оценивается сверху (например, 3). В то ж время очень трудно заменить их более подходящими пред положениями. Некоторый опыт показывает, что полученны на основе сделанных допущений результаты хорошо согла суются с практикой. Лишь допущение 3 сильно завышает оценку. Действительно, оно предполагает, что при $\rho \leqslant \alpha m$

номер $r_{\nu-\rho}$ в $\{\beta_0^{\nu,\rho}\}_{r_{\nu-\rho}}$ или в $\{\gamma_{r_\nu}^{\nu,\rho}\}_{r_{\nu-\rho}}$ не повторяется, т. е. начиная с $(\nu-\alpha m)$-й итерации и до ν-й направляющий элемент принадлежит различным строкам. Практически дело обстоит существенно иначе, так что действительное число ненулевых элементов $\beta_0^{\nu,\rho}$ или $\gamma_{r_\nu}^{\nu,\rho}$ меньше теоретического. Так как полнота $\varphi(\nu-\rho)$ η-вектора $\eta_{r_{\nu-\rho}}^{\nu-\rho}$ велика при малых ρ, то оценка завышается еще больше. Замена предположения 3 предположением о постепенном возрастании числа ненулевых элементов $\beta_0^{\nu,\rho}$ дает оценку снизу.

При сделанных допущениях можно рассчитать число умножений во всех алгоритмах и объем обрабатываемого числового материала. Вычислительные формулы здесь не приводятся, однако в следующем пункте представлены сравнительные результаты, полученные с их помощью.

5.5. Сравнение алгоритмов.

1. Число умножений.

Модифицированный алгоритм требует значительно меньшего числа умножений, чем остальные алгоритмы.

В типичном примере ($k = 2m$; $n = 2m$; $\pi = 0{,}03$; $\varphi = 0{,}30$; $\beta = 1$; два повторения в МА и ММА после m и $1{,}5\,m$ итераций; $\alpha_i = \alpha = 0{,}7$ для $i = 0, 1, 2$; $\varphi_i = 0{,}3$; $\gamma_i' = \gamma' = 0{,}15$; $\gamma'' = 0{,}25$ для всех i) получены следующие результаты (время решения прямым алгоритмом принято за 100):

ОМА = 56,1	1.	3,2
	2.	33,9
	3.	19,0
МА = 79,6	1.	21,2
	2.	35,7
	3.	19,0
	R.	3,7
ММА = 31,1	1.	16,1
	2.	8,5
	3.	4,8
	R.	1,7

Цифры 1, 2, 3 относятся к основным операциям. Операция 1 — вычисление главного столбца, операция 2 — преобразование обратной матрицы в ОМА, вычисление β_0^v $(\gamma_r \ldots)$ в МА (ММА), 3 — операция оценки, R — время на выполнение повторения. Отсюда можно сделать следующие выводы:

a. Когда алгоритм оценивается числом умножений, лучшим является модифицированный мультипликативный алгоритм, а за ним следует алгоритм с обратной матрицей.

b. Разница между прямым алгоритмом и алгоритмом с обратной матрицей уменьшается, когда убывает n, когда возрастает отношение π/φ либо когда возрастает α или β.

c. Повторения оказывают заметное влияние на общее число умножений. Для упомянутого примера имеют место следующие результаты:

	МА	ММА
Без повторений	136,6	62,6
Одно повторение после $^4/_3\, m$ итераций	91,2 (1,83)	38,0 (0,84)
Два повторения после m и $1{,}5\, m$ итераций	79,6 (3,66)	31,1 (1,68)
Три повторения после m, $^4/_3\, m$ и $^5/_3\, m$ итераций	73,1 (5,50)	27,9 (2,52)

В скобках указана часть умножений, приходящихся на долю повторений.

d. Если модифицировать алгоритм с обратной матрицей и преобразовывать d-строку на каждой итерации, вычисляя направляющую строку, число умножений в ОМА может быть снижено до 41,9.

2. Вспомогательные операции.

Как мы видели в п. 5.2, в ПА требуется большое число вспомогательных операций, не возникающих или почти не возникающих в других алгоритмах. Это делает прямой алгоритм еще менее привлекательным.

3. Объем обрабатываемого числового материала.

Замечая, что в ПА мы должны считывать и перезаписывать преобразованную матрицу задачи на каждой итерации, в ОМА мы должны считывать и перезаписывать обратную

матрицу базиса и считывать исходную матрицу в течение каждой итерации, в МА и ММА мы должны считывать η-векторы и исходную матрицу (неиспользованные столбцы в ММА) в течение каждой итерации (необходимой записью можно пренебречь), получим следующие значения объема Q обрабатываемого числового материала.

Если

в ПА $Q_r = Q_w = 100$ (Q_r — количество считываемых, Q_w — количество записываемых чисел),

то

в ОМА $Q_r = 45{,}5$ и $Q_w = 32{,}6$

в МА без повторений	$Q_r = 106{,}0$	
одно повторение ($^4/_3\, m$)	$Q_r = 82{,}0$	
два повторения (m; $1{,}5\, m$)	$Q_r = 76{,}9$	
три повторения (m; $^4/_3\, m$; $^5/_3\, m$)	$Q_r = 74{,}7$	
в ММА без повторений	$Q_r = 95{,}0$	
одно повторение ($^4/_3\, m$)	$Q_r = 66{,}9$	$Q_w = 2{,}0$
два повторения (m; $1{,}5\, m$)	$Q_r = 58{,}8$	$Q_w = 4{,}0$
три повторения (m; $^4/_3\, m$; $^5/_3\, m$)	$Q_r = 55{,}8$	$Q_w = 5{,}9$

Заметим, что во время повторений в ММА производятся операции записи, поскольку преобразование столбцов делается как в прямом алгоритме.

Мы видим, что снова модифицированный мультипликативный алгоритм является наилучшим, но теперь за ним следует мультипликативный алгоритм.

Эти результаты не удивительны, если учесть, что мультипликативный алгоритм был разработан для сокращения числа обращений к внешней памяти, особенно для сокращения числа перезаписей.

Заметим также, что число обращений к внешней памяти убывает при повторениях. Те же результаты получены в ряде других примеров.

4. Точность.

При удвоенной разрядности преимущества алгоритмов, использующих обратную матрицу, увеличиваются, так как в ПА все вычисления приходится производить с удвоенной разрядностью.

Предполагая, что в ОМА и МА мы выполняем два повторения с удвоенной точностью и что одно умножение

с удвоенной точностью эквивалентно трем с обычной, получим следующую картину для числа умножений в рассматриваемом примере.

ПА	300		
ОМА	88,5	1.	3,2
		2.	33,9
		3.	19,0
		R.	32,4
МА (2 повторения)	87,0	1.	21,2
		2.	35,7
		3.	19,0
		R.	11,1
ММА (2 повторения).	34,6	1.	16,1
		2.	8,5
		3.	4,8
		R.	5,1

Преимущества модифицированного мультипликативного алгоритма очевидны, мультипликативный алгоритм — на втором месте. При хранении чисел с удвоенной разрядностью объем обрабатываемого числового материала также возрастает. Это дает еще один аргумент в пользу модифицированного мультипликативного алгоритма. Заметим, что даже при обычной разрядности чисел благодаря повторениям МА и ММА дают более точные результаты.

5. Гибкость программы.

Ясно, что благодаря особенностям повторения в МА и ММА легко возобновить вычисления. Если из машины извлечен список переменных текущего базиса, можно возобновить вычисления, начав с повторения.

Дополнительное преимущество ММА состоит в том, что при этом будет получено множество η-векторов с малым числом ненулевых элементов. Так как повторение не требует большого времени, едва ли необходимо специально корректировать программу.

Недостатком мультипликативного алгоритма является трудность внесения изменений в оптимизируемую функцию в процессе вычислений, так как они влекут за собой изменение „нулевых" элементов всех η-векторов. В модифицированном мультипликативном алгоритме всегда можно пере-

считать d-строку без каких-либо изменений η-векторов. Поэтому новый алгоритм более гибок. Наконец, в п. 4.6 мы видели, что модифицированный мультипликативный алгоритм можно использовать как вычислительный алгоритм двойственного симплексного метода.

6. Сложность программы.

Ясно, что прямой алгоритм требует наиболее простой программы и сложность программ возрастает в порядке ПА, ОМА, МА и ММА. Последний алгоритм может быть слишком сложным для малых машин. Для таких машин предпочтительнее алгоритм с обратной матрицей.

Заключение.

1. Лучшим алгоритмом для больших вычислительных машин является модифицированный мультипликативный алгоритм.

Основания:

Этот алгоритм требует меньшего, чем другие алгоритмы, числа умножений, требует меньшего числа пересылок между лентой и барабаном, допускает частичную работу с удвоенной разрядностью (при повторениях) без значительного возрастания времени решения, позволяет легко возобновлять решение.

2. Для машин средних размеров предпочтительнее алгоритм с обратной матрицей.

Основания:

Этот алгоритм обычно требует меньшего числа умножений, чем мультипликативный алгоритм, его программа проще программы мультипликативного алгоритма. Поскольку размеры задач ограничены, трудно ожидать значительных ошибок округления при обычной разрядности, так что соответствующее преимущество мультипликативного алгоритма не имеет серьезного значения.

3. Наконец, для малых вычислительных машин самым подходящим представляется прямой алгоритм, поскольку его программа самая простая.

Глава 6

НЕКОТОРЫЕ СПЕЦИАЛЬНЫЕ ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

6.1. Введение.

В этой главе рассматриваются некоторые специаль задачи линейного программирования, возникающие при ре нии нелинейных задач, описанных в гл. 7—11.

Пункт 6.2 посвящен рассмотрению задач с двусторонн ограничениями на переменные, в п. 6.3 рассматривается дача минимизации суммы модулей переменных, называе в дальнейшем задачей с абсолютными значениями.

6.2. Задачи с двусторонними ограничениями на переменные.

В некоторых случаях переменные в задаче линейно программирования бывают ограничены как снизу, так и свер Такую задачу можно представить в виде

$$\max\{p^T x \mid Ax + y = b,\ 0 \leqslant x \leqslant c_1,\ 0 \leqslant y \leqslant c_2\}, \quad (6.2.$$

где c_1 и c_2 — неотрицательные n- и m-мерные векторы со ветственно (некоторые из их компонент могут быть неогра ченными). Задачу можно решить введением дополнительн переменных и ограничений в виде равенств, однако при эт размеры задачи увеличиваются и возрастает время решен

Чарнсом и Лемке [37] и Данцигом [14] было показа что можно оперировать с таблицей размеров m на n, ук зывая для каждой внебазисной переменной, достигает ли о своей верхней границы (т. е. при $x_j + \bar{x}_j = c_{1j}$ мы ра сматриваем либо x_j, либо $\bar{x}_j$; эти переменные имеют общ верхнюю грань, и если $x_j = c_{1j}$, то в качестве внебазисн рассматривается переменная $\bar{x}_j$). Главный столбец выбирает как обычно. Однако выбор направляющего элемента бол

сложен. Пусть на некоторой итерации базисные переменные обозначены y_i^*, их значения b_i^*, верхние границы $c(y_i^*)$. Если в базис вводится переменная x_s^*, которой соответствуют столбец $\bar{a}^*_{.s}$ и верхняя граница $c(x_s^*)$, то величина x_s^* может быть выбрана в пределах от нуля до λ^*, где

$$\lambda^* = \min\{\lambda_1^*,\ \lambda_2^*,\ \lambda_3^*\} \quad (6.2.2)$$

и

$$\lambda_1^* = \max\{\lambda \mid b^* - \lambda \bar{a}^*_{.s} \geqslant 0\}, \quad (6.2.3)$$

$$\lambda_2^* = c(x_s^*), \quad (6.2.4)$$

$$\lambda_3^* = \max\{\lambda \mid b^* - \lambda \bar{a}^*_{.s} \leqslant c(y^*)\}, \quad (6.2.5)$$

где $c(y^*)$ — вектор с компонентами $c(y_i^*)$.

Если в исходном базисе ни одно значение переменной не превосходит верхней границы, то указанный способ выбора гарантирует, что это никогда не произойдет. Если $\lambda^* = \lambda_1^*$, то формулы преобразования те же, что и в симплексном методе. Если $\lambda^* = \lambda_2^*$, положим x_s^* равным его верхней границе, т. е. заменим вектор b^* вектором $b^* - c(x_s^*)\bar{a}^*_{.s}$, затем $\bar{a}^*_{.s}$ и $\bar{d}_s^*$ заменим соответственно векторами $-\bar{a}^*_{.s}$ и $-\bar{d}_s^*$ и будем рассматривать $\bar{x}_s^*$ как внебазисную переменную вместо x_s^*. В этом случае не требуется дальнейших преобразований. Если $\lambda^* = \lambda_3^*$ и выбирается строка r, то y_r^* исключается из базиса. После обычных преобразований с отрицательным направляющим элементом положим y_r^* равным его верхней границе.

Действуя таким образом, мы получим оптимальное решение после конечного числа шагов. Это число не уменьшается от применения описанного способа решения задачи с двусторонними ограничениями. Однако некоторые преобразования весьма просты (при $\lambda^* = \lambda_2^*$), так как не делается преобразований матрицы или обратной матрицы базиса, операций оценки, не вычисляется $\beta_0^{\nu}(\gamma_{r_\nu}^{\nu})$, а также не строится новый η-вектор. Весьма полезно при определении главного столбца выбрать несколько отрицательных значений d_j и соответствующих им столбцов. Тогда при выборе следующего

главного столбца в случае $\lambda^* = \lambda_2^*$ устраняются опер[ации] поиска на магнитной ленте.

Вагнер [7] отметил, что для решения задач с двус[то]ронними ограничениями на переменные лучше использов[ать] двойственный симплексный метод. Дело в том, что е[сли] начать решение, положив все переменные, для которых [...] положительны, равными их верхним границам, то устр[а]няется первая стадия решения. В этом случае элемент[ы] исходной d-строки неотрицательны. Критерием выбора гла[в]ной строки будет либо значение $b_r^* < 0$, либо $c(y_r^*) - b_r^* <$ [...] (обычно среди них выбирается строка с наименьшим знач[е]нием). В первом случае можно выбрать направляющий эл[е]мент по обычным правилам и произвести преобразовани[я]; во втором случае умножаем строку r на -1, b_r^* заме[]няем $c(y_r^*) - b_r^*$ и $y_r^* - \bar{y}_r^*$. Затем можно обычным путе[м] выбрать направляющий элемент и произвести преобразо[ва]ния. Если значение переменной, которую следует ввести в базис, превзойдет $c(x_s^*)$ (так что, может быть, ее при[]дется исключить из базиса на следующей итерации), то выгоднее, как упоминалось выше, не выбирать $\bar{a}_{rs}^*$ направляющим элементом, приравнять x_s^* его верхней границе и выбрать по обычным правилам второй направляющий элемент в той же строке, используя обычные критерии, но пренебрегая s-м столбцом. Если значение нового элемента, вводимого в базис, снова превзойдет его верхнюю границу, мы отвергаем также и его, приравниваем его значение верхней границе, выбираем третий направляющий элемент той же строки и т. д. При этих операциях появляются отрицательные значения d_j, но, как легко проверить, после выбора направляющего элемента в этой строке и преобразований все d_j снова неотрицательны.

Решение задачи описанным методом требует конечного числа шагов. Некоторые итерации очень просты. Вычисление главного столбца — единственная основная операция при решении задачи с помощью двойственного алгоритма с обратной матрицей или двойственного модифицированного мультипликативного алгоритма. Операции поиска на ленте в значительной мере сокращаются выбором нескольких предполагаемых главных столбцов.

В качестве примера приведем задачу, встречающуюся в дальнейшем:

$$\max\{p^T x \mid A_1 x \leqslant 0;\quad A_2 x = 0;\quad -1 \leqslant x_j \leqslant 1 \ \text{при}\ j \in J_1;\quad 0 \leqslant x_j \leqslant 1 \ \text{при}\ j \in J_2;\quad -1 \leqslant x_j \leqslant 0 \ \text{при}\ j \in J_3;\quad x_j = 0 \ \text{при}\ j \in J_4\},\tag{6.2.6}$$

где $J_1 + J_2 + J_3 + J_4 = J = \{1, \ldots, n\}$, A_1 и A_2 — матрицы $m_1 \times n$ и $m_2 \times n$ соответственно.

В этой задаче некоторые переменные ограничены снизу величиной, отличной от нуля. Такие переменные выгодно увеличивать, если соответствующие значения d_j меньше нуля, и уменьшать, когда d_j больше нуля.

Записав условия в виде $A_1 x + y = 0$, $y \geqslant 0$, и $A_2 x + z = 0$, $z = 0$, получим новые переменные y_i, не ограниченные сверху, и искусственные переменные z_i. Эту задачу можно решать способом, основанным на симплексном методе; мы, однако, опишем способ решения задачи с двусторонними ограничениями, основанный на двойственном симплексном методе.

Будем действовать следующим образом:

1. Если $j \in J_1$ и $p_j \geqslant 0$, положим x_j равным его верхней границе (т. е. вычитаем $a_{\cdot j}$ из b и $a_{\cdot j}$ заменяем $-a_{\cdot j}$);

если $j \in J_1$ и $p_j < 0$, положим x_j равным его нижней границе (т. е. добавляем $a_{\cdot j}$ к b);

если $j \in J_2$ и $p_j \geqslant 0$, положим x_j равным его верхней границе;

если $j \in J_3$ и $p_j \geqslant 0$, x_j заменяем $-x_j$ (т. е. $a_{\cdot j}$ на $-a_{\cdot j}$);

если $j \in J_3$ и $p_j < 0$, положим x_j равным его нижней границе. Переменные x_j, $j \in J_4$, во внимание не принимаются (их можно исключить из рассмотрения).

С этого момента все переменные должны быть неотрицательны, а верхние границы равны $c(x_j) = 2$ при $j \in J_1$, $c(x_j) = 1$ при $j \in J_2 + J_3$.

2. Пользуясь описанным методом выбора направляющего элемента в строке (не выбирая некоторые из них, когда это возможно) и пренебрегая столбцами $j \in J_4$, исключим из базиса искусственные векторы.

3. Применяем способ решения задачи с двусторонними ограничениями, основанный на двойственном симплексном методе.

6.3. Задача с абсолютными значениями.

Рассматривая задачу, двойственную к (6.2.6) (см. теорему 8 п. 2.5), получаем

$$\min\left\{\sum_{j\in J_1} v_j^+ + \sum_{j\in J_1} v_j^- + \sum_{j\in J_2} v_j^+ + \sum_{j\in J_3} v_j^- \,\middle|\, A_1^T u_1 + A_2^T u_2 + v^+ - v^- = p,\ u_1 \geqslant 0,\ v^+ \geqslant 0,\ v^- \geqslant 0\right\} \tag{6.3.1}$$

или, полагая $v_j = v_j^+ - v_j^-$ (так что v_j — свободная переменная), получаем

$$\max\left\{-\sum_{j\in J_1} |v_j| - \sum_{j\in J_2} \max\{v_j, 0\} + \sum_{j\in J_3} \min\{v_j, 0\} \,\middle|\, A_1^T u_1 + A_2^T u_2 + v = p,\ u_1 \geqslant 0\right\}. \tag{6.3.2}$$

Вектор u_1 состоит из переменных u_{1i}, $i\in I_1$, вектор u_2 — из переменных u_{2i}, $i \in I_2$.

Обозначим $u(I_1) = \{u_i \mid i \in I_1\}$ — множество всех переменных u_{1i}. Аналогично определим $u(I_2)$, $v(J_1)$, $v(J_2)$, $v(J_3)$ и $v(J_4)$. Заметим, что теперь строки обозначены индексом j, а столбцы — i. Главному столбцу припишем индекс r, главной строке — s. Элементы матрицы (A_1^T, A_2^T, E) обозначим $\bar{a}_{ji}$.

Если множества J_2, J_3 и J_4 пусты, так что $J_1 = J$, задача сводится к минимизации суммы абсолютных значений „невязок“ системы линейных неравенств. Поэтому будем называть ее задачей с абсолютными значениями.

Для решения этой задачи можно предложить способ, тесно примыкающий к способу, описанному в предыдущем пункте. Так как переменные v_j свободны, то, положив $v = p$, получим исходный базис. Рассчитаем затем исходную d-строку, используя следующие коэффициенты оптимизируемой функции

при v_j:

$$q(v_j) = \begin{cases} -1, & \text{если } p_j \geqslant 0, \\ +1, & \text{если } p_j < 0, \end{cases} \quad j \in J_1,$$

$$q(v_j) = \begin{cases} -1, & \text{если } p_j \geqslant 0, \\ +0, & \text{если } p_j < 0, \end{cases} \quad j \in J_2,$$

$$q(v_j) = \begin{cases} -0, & \text{если } p_j \geqslant 0, \\ +1, & \text{если } p_j < 0, \end{cases} \quad j \in J_3,$$

$$q(v_j) = 0, \qquad j \in J_4.$$

Таким образом, в соответствии с (6.3.1) при $p_j \geqslant 0$ базису принадлежит v_j^+, если $p_j < 0$, то в базис входит $-v_j^-$.

Для дальнейшего положим также $q(u_{1i}) = -0$ для всех $i \in I_1$ и $q(u_{1i}) = 0$, если $i \in I_2$ (знак в этом случае не имеет значения). Тогда при $p_j \neq 0$ и $j \notin J_4$ величины p_j и $q(v_j)$ всегда противоположны по знаку (учитывая и знаки нуля).

Пусть

$$\delta(v_j) = 2 \text{ при } j \in J_1, \quad \delta(v_j) = 1 \text{ при } j \in J_2 + J_3. \tag{6.3.3}$$

Предположим, что проведено некоторое количество итераций. Полученные значения обозначим звездочкой. В предлагаемой методике всегда справедливы следующие два условия:

1. Для внебазисных переменных оценка $d^*(v_j)$ совпадает с $d^*(v_j^+)$. Тогда

$$d^*(v_j^-) = \delta(v_j) - d^*(v_j^+). \tag{6.3.4}$$

2. Если v_j принадлежит базису, то при $p^*(v_j) > 0$ это v_j^+, соответствующий коэффициент оптимизируемой функции $q(v_j^+)$; при $p^*(v_j) < 0$ это $-v_j^-$, соответствующий коэффициент оптимизируемой функции $q(-v_j^-) = q(v_j^+) + \delta(v_j)$; при $p^*(v_j) = 0$ это либо v_j^+, либо $-v_j^-$.

В качестве главного столбца мы можем теперь выбрать

a. Произвольный внебазисный столбец, соответствующий u_{1i}, если $d^*(u_{1i}) < 0$.

b. Произвольный внебазисный столбец, соответствующий u_{2i}, если $d^*(u_{2i}) \neq 0$.

c. Произвольный внебазисный столбец, соответствующий v_j, если $d^*(v_j) < 0$.

d. Произвольный внебазисный столбец, соответствующий v_j, если $d^*(v_j) > \delta(v_j)$.

Если таких столбцов нет, задача решена.

Обычно мы будем начинать решение, последовательно вводя в базис u_{2l}. Если все u_{2l} принадлежат базису, выбираем главным „лучший" столбец среди удовлетворяющих условиям a — d. Пусть r — номер выбранного главного столбца и u_r^* — переменная, вводимая в базис (быть может переменная v_j). Если u_r^* — переменная v_j и $d^*(v_j) > \delta(v_j)$, заменим $d^*(v_j)$ на $\delta(v_j) - d^*(v_j)$. При этом вместо v_j^- в базис вводится $-v_j^-$ $(u_r^* = -v_j^-)$.

Направляющий элемент главного столбца должен удовлетворять требованиям:

1. $\operatorname{sgn} q(v_j) = -\operatorname{sgn} p^*(v_j)$ для всех базисных v_j, $j \notin I$ и $p^*(v_j) \neq 0$.

2. Если переменная v_j исключается из базиса, ее оценка должна лежать между нулем и $\delta(v_j)$. Если это условие не выполняется для переменной v_j, соответствующей выбранной главной строке, выбирается другой направляющий элемент.

Эти два требования приводят к следующим операциям:

1. Если $u_r^* \in u(I_1)$, $v^+(J_1)$, $v^+(J_2)$ или $v^+(J_3)$ либо если $u_r^* \in u(I_2)$ и $d(u_r^*) \leqslant 0$, то s определяется условиями

$$\frac{p_s^*}{\bar a_{sr}^*} = \min_j \left\{ \frac{p_j^*}{\bar a_{jr}^*} \,\middle|\, \bar a_{jr}^* \neq 0,\ \operatorname{sgn} q(v_j^*) = -\operatorname{sgn} \bar a_{jr}^*;\quad v_j^* \notin u(I_2) + v(J_4) \right\}.$$

Если $-u_r^* \in v^-(J_1)$, $v^-(J_2)$ или $v^-(J_3)$ либо если $u_r^* \in u(I$ и $d(u_r^*) > 0$, то s определяется условиями

$$\frac{p_s^*}{-\bar a_{sr}^*} = \min_j \left\{ \frac{p_j^*}{-\bar a_{jr}^*} \,\middle|\, \bar a_{jr}^* \neq 0,\ \operatorname{sgn} q(v_j^*) = +\operatorname{sgn} \bar a_{jr}^*;\quad v_j^* \notin u(I_2) + v(J_4) \right\}.$$

2. Если $v_s^* \in u(I_1)$, производим преобразования. Если $v_s^* \in v(J_1) + v(J_2) + v(J_3)$, вычисляем $d_r^* + \delta(v_s^*)|\bar a_{sr}^*|$. Если

$d_r^* + \delta(v_s^*)|\bar a_{sr}^*| \geqslant 0$, производим преобразования, если же $d_r^* + \delta(v_s^*)|\bar a_{sr}^*| < 0$, то

a. $\bar a_{sr}^*$ не выбирается направляющим элементом.

b. Заменяем $q(v_s^*)$ на $q(v_s^*) - \{\delta(v_s^*) + 0\} \operatorname{sgn} q(v_s^*)$ [член $+0$ гарантирует желаемый знак нового $q(v_s^*)$].

c. Заменяем

$$d_i^* \text{ на } d_i^* + \delta(v_s^*)\, \bar a_{si}^* \operatorname{sgn} \bar a_{sr}^*,$$

$$d_0^* \text{ на } d_0^* + \delta(v_s^*) \cdot p_s^* \cdot \operatorname{sgn} \bar a_{sr}^*$$

(d_0^* — значение оптимизируемой функции).

d. Повторяем шаги 1 и 2.

Отбрасывание направляющего элемента приведет в конце концов к преобразованию с другим направляющим элементом. При этом может измениться знак p^* в отвергнутой строке. В невырожденном случае знак изменяется всегда, при вырождении значение p^* может обратиться в нуль после преобразования. В первом случае для соблюдения условия 1 надо после преобразования изменить коэффициент $q^*(v_j)$ при переменной, соответствующей отвергаемой строке. Такое изменение не приносит никакого вреда и в последнем случае. Поэтому мы выполняем операцию 2b. В результате изменится вся d-строка.

Новое значение $d(u_r^*)$ будет все еще отрицательным, так что возможен выбор второго направляющего элемента в этом столбце. Поскольку для отвергнутых направляющих элементов знаки q и p^* более не соответствуют друг другу, мы автоматически исключим эти переменные из рассмотрения при выборе второго направляющего элемента. При таком процессе вычислений вводимая в базис переменная v_j^+ будет всегда неотрицательной, а переменная $-v_j^-$ — неположительной. Ясно, что направляющий элемент в главном столбце будет выбран после конечного числа отбрасываний (заметим, что в этой задаче максимальное значение оптимизируемой функции всегда не больше нуля, так что неограниченного решения быть не может). Затем в результате преобразования условие 1 будет удовлетворено. Если переменная v_j^- покидает базис, меняем после преобразования ее оценку $-d(v_j^-)$ на $d(v_j^+)$.

т. е. прибавляем $\delta(v_j)$ к $-d(v_j^-)$. Следовательно, требова-
ние 2 также удовлетворяется.

Действуя таким способом, можно решить задачу (6.3.1).
Этот способ по существу является симплексным методом,
состоящим из нескольких весьма простых итераций. Если вы-
рожденность преодолена, задача решается за конечное чис-
шагов.

Поскольку задача (6.3.1) является двойственной к (6.2.6),
мы на основании теорем 7 и 8 п. 2.5 сразу получаем реше-
ние (6.2.6):

$$x_j = -d'(v_j^+) - q(v_j^+), \qquad (6.3.5)$$

$$y_i = d'(u_{1i}) \qquad (6.3.6)$$

(штрих означает оптимальное решение), так что

$$x_j = \begin{cases} +1, \text{ если } v_j^+ \text{ принадлежит конечному базису и } j \in J_1 + J_2, \\ 0, \text{ если } v_j^+ \text{ принадлежит конечному базису и } j \in J_3, \\ 0, \text{ если } v_j^- \text{ принадлежит конечному базису и } j \in J_3, \\ -1, \text{ если } v_j^- \text{ принадлежит конечному базису и } j \in J_1 + J_3, \\ 0, \text{ если } j \in J_4. \end{cases}$$

Для решения задачи с абсолютными значениями легко при-
менить модифицированный мультипликативный алгоритм.
Операция нахождения направляющего элемента в главном
столбце будет более сложной, чем аналогичная операция,
описанная в п. 4.5. Но эта операция не основная. В слу-
чае когда отвергается направляющий элемент, изменение
d-строки можно производить на следующей операции вы-
числения γ. Нам придется лишь начинать эту операцию
с вектора $\gamma_{s_\nu}^{\nu,1}$, определяемого условиями $\{\gamma_{s_\nu}^{\nu,1}\}_{s_\nu} = d^\nu(v_{s_\nu}^{\nu-1})$,
где $v_{s_\nu}^{\nu-1}$ — переменная, покидающая последней базис на $(\nu-1)$-й
итерации. Если эта переменная v_j^-, то имеется в виду значение
$-d^\nu(v_j^-)$ [до замены его на $d^\nu(v_j^+)$].

$$\{\gamma_{s_\nu}^{\nu,1}\}_j = \begin{cases} \delta(v_j^{\nu-1}) \operatorname{sgn} \bar{a}_{jr_\nu}^{\nu-1}, \text{ если } v_j^{\nu-1} \text{ — отвергнутая переменная}, \\ 0 \quad \text{в остальных случаях.} \end{cases}$$

Во время операции вычисления γ значения $d^\nu(v_j^+)$ вычисляются совместно с оценками $d^\nu(u_{1i})$ использованных переменных u_{1i}. Заметим, что, когда переменная v_j покидает базис, ее η-вектор всегда соответствует переменной v_j^+. Величина $d^\nu(v_j^-)$ может быть легко получена из $d^\nu(v_j^+)$.

Из предыдущего следует, что итерации, на которых отвергается направляющий элемент, весьма просты. Они приводят лишь к возрастанию числа ненулевых элементов исходного вектора при вычислении γ на следующей полной итерации.

Задача с абсолютными значениями возникает как вспомогательная задача в некоторых методах нелинейного программирования, описанных в гл. 7—11. В этих методах приходится решать последовательность подобных задач, причем соседние задачи могут отличаться друг от друга в следующем:

1. Множества J_1, J_2, J_3, J_4 переходят в $\tilde{J}_1$, $\tilde{J}_2$, $\tilde{J}_3$, $\tilde{J}_4$, однако $\tilde{J}_1 + \tilde{J}_2 + \tilde{J}_3 + \tilde{J}_4 = J$ (эти множества никогда не пересекаются).

2. Множества I_1 и I_2 переходят в $\tilde{I}_1$ и $\tilde{I}_2$. Здесь встречаются лишь следующие варианты:

a. $i \in I_2$ и $i \in \tilde{I}_1$.

b. $i \in I_1$ и $i \notin \tilde{I}_1$ (только когда u_{1i} не входит в конечный базис).

c. $i \notin I_1 + I_2$ и $i \in \tilde{I}_1$ [тогда новое значение $d(u_i) < 0$].

d. $i \notin I_1 + I_2$ и $i \in \tilde{I}_2$.

3. Вектор p заменяется вектором $\tilde{p}$.

4. Для некоторых значений i вектор $a_{\cdot i}$ может быть заменен вектором $\tilde{a}_{\cdot i}$.

Покажем теперь, как использовать решение и множество η-векторов предыдущей задачи в начале решения следующей. Задача решается с помощью модифицированного мультипликативного алгоритма. Штрихом обозначены величины оптимального решения предшествующей задачи.

1. Эти изменения просты. Вводятся новые значения $q(v_j)$ и $\delta(v_j)$. Если вектор p не изменяется при переходе к следующей задаче, коэффициенты оптимизируемой функции

определяются с помощью вектора p', а не p. В противном случае сначала выполняются операции 3.

Необходим пересчет строки d.

2.a. Заменяем I_1, I_2 на $\tilde{I}_1$ и $\tilde{I}_2$. В дальнейшем при выборе вектора, покидающего базис, переменная u_l из рассмотрения не исключается. Если ее базисное значение отрицательно, $p'(u_l) < 0$, припишем ей большой „штрафной" коэффициент $+M$ в оптимизируемой функции, $q(u_l) = +M$ (достаточно большой, чтобы переменная с отрицательным значением не оставалась в базисе). Положим далее $\delta(u_l) = M$ и при выборе направляющего элемента будем рассматривать u_l так же, как и переменные v_j. Тогда переменная u_l либо покинет базис на одной из последующих итераций, после чего коэффициент M стирается, либо останется в базисе (направляющий элемент отвергнут), причем $p(u_l) \geqslant 0$. В этом случае необходим пересчет значений d.

b. Необходимо лишь опустить столбец $a_{\cdot l}$ исходной матрицы и оценку $d(u_l)$ в d-строке.

c, d. Необходимо добавить столбец $a_{\cdot l}$ к исходной матрице, преобразовать его с помощью η-векторов и получить его оценку d.

3. Вычисляем $\tilde{p}'$ с помощью η-векторов.

Могут встретиться два случая.

(I) $\tilde{p}'(u_{1l}) < 0$ для некоторого l. Приписываем тогда u_{1l} „штрафной" коэффициент $+M$ в оптимизируемой функции, $q(u_{1l}) = +M$, вводим $\delta(u_{1l}) = M$, при выборе направляющего элемента рассматриваем u_{1l} так же, как и переменные v_j.

(II) Условие $\operatorname{sgn} q(v_j)\, \tilde{p}'(v_j) \leqslant 0$ нарушается для некоторого j. Заменим тогда $q(v_j)$ на $q(v_j) - \{\delta(v_j) + 0\} \operatorname{sgn} q(v_j)$. В обоих случаях необходим пересчет значений d.

4. Если столбец $a_{\cdot l}$ не входит в оптимальный базис, легко заменить его в исходной матрице на $\tilde{a}_{\cdot l}$ и пересчитать соответствующую оценку d.

Если $a_{\cdot l}$ принадлежит оптимальному базису, добавляем $\tilde{a}_{\cdot l}$ к исходной матрице, вычеркиваем $a_{\cdot l}$ из нее, вводим переменную $\tilde{u}_l$, а u_l считаем искусственной. [Приписываем ей „штрафной" коэффициент $-M$ при $\tilde{p}(u_l) \geqslant 0$ и $+M$ при $\tilde{p}(u_l) < 0$; такая переменная должна быть исключена из ба-

зиса как можно скорее, поэтому соответствующий направляющий элемент не отвергается.]

Ясно, что если изменений типа 3 и 4 слишком много, становится трудно и даже неразумно начинать решение, отправляясь от оптимального решения предыдущей задачи. Можно, однако, начать решение с предварительных операций, как описано в п. 4 и 5. Список переменных, выбираемых для введения в базис на предварительных операциях, можно извлечь из результатов решения предыдущей задачи.

Конечно, таким способом можно воспользоваться, даже когда две задачи мало сходны и проведение предварительных операций, по-видимому, невыгодно.

Глава 7

МЕТОДЫ ВОЗМОЖНЫХ НАПРАВЛЕНИЙ

7.1. Введение.

Пусть $F(x)$ — вогнутая функция n-мерного вектора $x \in$ $f_i(x)$ — выпуклые функции $x \in E_n$ при $i \in I_C$, $I_C \subset I = \{1, \ldots,$ a_i — n-мерные векторы для $i \in I_L = I - I_C$. Пусть дале и p — n-мерные векторы, b — m-мерный вектор (некотор или все компоненты вектора c могут быть неограниченны

Рассмотрим теперь задачу

$$\max \{F(x) \mid f_i(x) \leqslant b_i, \quad i \in I_C; \ a_i^T x \leqslant b_i, \quad i \in I_L; \ 0 \leqslant x \leqslant$$ (7.1.

Область R, определяемая ограничениями (7.1.1), выпук Задача (7.1.1) называется задачей *выпуклого программир вания*. Будем предполагать, что нелинейные ограничен $f_i(x) \leqslant b_i$ удовлетворяют условиям регулярности С1 п. 2 Будем также считать функции $F(x)$ и $f_i(x)$ дифференциру мыми с непрерывными частными производными. Положим

$$g(x) = \left\{ \frac{\partial F}{\partial x_j}, \ j = 1, \ldots, n \right\},$$ (7.1

$$q_i(x) = \left\{ \frac{\partial f_i}{\partial x_j}, \ j = 1, \ldots, n \right\}, \quad i \in I_C.$$ (7.1

Введем в рассмотрение неотрицательные переменные определяемые условиями

$$y_i = y_i(x) = \begin{cases} b_i - f_i(x) & \text{при } i \in I_C, \\ b_i - a_i^T x & \text{при } i \in I_L. \end{cases}$$ (7.1.

Пусть m — максимум $F(x)$ на R. Предположим, чт либо вогнутая функция $F(x)$ не ограничена на R ($m = \infty$ либо множество $R_m = \{x \in R \mid F(x) = m\}$ ограничено (усло вие С3).

Теорема 1. *Если $F(x)$ удовлетворяет на R усло вию С3 и $m < \infty$, то $R_\alpha = \{x \in R \mid F(x) \geqslant \alpha\}$ ограничен для любого α.*

Доказательство. $R_\alpha = 0$ при $\alpha > m$. Множество R_α содержит R_m и выпукло при $\alpha \leqslant m$ (п. 2.4, теорема 8). Пусть R_α не ограничено для некоторого α, тогда для любого $x \in R_m$ найдется вектор s, такой, что $x + \lambda s \in R_\alpha$ для всех $\lambda > 0$. Выберем настолько большое λ_1, что $x_1 = x + \lambda_1 s \notin R_m$. Тогда $F(x_1) < m = F(x)$ и в силу следствия 1 теоремы 3 п. 2.4 $g(x_1)^T (x - x_1) > 0$. Следовательно, $g(x_1)^T s < 0$. Заметим, что $x + \lambda s = x_1 + (\lambda - \lambda_1) s$ и $F(x + \lambda s) \leqslant F(x_1) + (\lambda - \lambda_1) g(x_1)^T s < \alpha$, если λ — достаточно большое число. Следовательно, включение $x + \lambda s \in R_\alpha$ не может иметь места при всех $\lambda > 0$.

Следствие. Если $m < \infty$, то для любого $y \in R$ множество $\{x \in R \mid F(x) \geqslant F(y)\}$ ограничено.

Метод решения задачи выпуклого программирования (7.1.1) будем называть *методом возможных направлений*, если он обладает следующими свойствами:

1. Исходная точка принадлежит допустимой области, $x^0 \in R$.

2. Построив точки $x^0, x^1, \ldots, x^k \in R$, удовлетворяющие условию $F(x^h) > F(x^{h-1})$, $h = 1, \ldots, k$, определяем точку $x^{k+1} \in R$, $F(x^{k+1}) > F(x^k)$, либо обнаруживаем, что такой точки не существует и x^k — максимальное решение (7.1.1), либо $F(x)$ не ограничен. Процесс определения x^{k+1} из R состоит из двух частей.

a. В точке x^k определяется подходящее возможное направление s^k (определение возможного и подходящего возможного направления см. в п. 2.6).

b. Определяется длина шага λ_k, такая, что $x^{k+1} = x^k + \lambda_k s^k \in R$ и $F(x^{k+1}) > F(x^k)$.

3. Выбор направлений s^k и скаляров λ_k, т. е. точек $x^k \in R$, производится так, что либо обеспечивается сходимость $F(x)$ к m

$$\lim_{k \to \infty} F(x^k) = m, \tag{7.1.5}$$

либо последовательность $F(x^k)$ не ограничена. В последнем случае решение задачи (7.1.1) не ограничено.

Методы возможных направлений можно рассматри
как градиентные методы с большим шагом [1]). Их следу
отличать от градиентных методов с малым шагом [1]), рассм
ренных, например, Эрроу, Гурвицем и Удзавой [40, гл. 6—
Многие методы решения линейных и нелинейных за
программирования можно трактовать как методы возможн
направлений. Они различаются лишь дополнительными тр
бованиями к выбору исходной точки x^0, направления s^k и
длины шага λ_k. К таким методам относятся симплекс
метод, метод одновременного решения прямой и двойстве
ной задач, метод проекций градиента Розена [28], мет
квадратичного программирования Била [4, 5]; метод Фран
и Вулфа [32] нелинейного программирования при линейн
ограничениях. В п. 7.2 изучается задача отыскания исход
точки x^0; п. 7.3 и гл. 8 посвящены различным путям выбо
подходящего возможного направления. Определение дли
шага рассматривается в п. 7.4. В п. 7.5 описываются алгоритм
Сходимость итерационного процесса изучается в п. 7.6. На
нец, в п. 7.7 обсуждается роль предположений о вогн
тости $F(x)$ и выпуклости $f_i(x)$. Гл. 9 посвящена зада
линейного программирования, гл. 10 — задаче квадратичн
программирования, гл. 11 — задаче максимизации вогнут
функции при линейных либо при линейных и нелинейн
ограничениях, определяющих выпуклую область. В эт
главах рассматривается также, как может быть улучше
сходимость итерационного процесса.

7.2. Определение исходного возможного решения.

Во многих практических задачах точку $x^0 \in R$ легко п
лучить из физических соображений. Если же возможное и
ходное решение $x^0 \in R$ неизвестно, выберем произволь
точку x^0, удовлетворяющую условиям $0 \leqslant x^0 \leqslant c$. Вве
затем дополнительную переменную ξ_1 и неотрицатель
числа ρ_i, $i \in I_L$, такие, что $\rho_i = 0$ при $y_i(x^0) \geqslant 0$ и $\rho_i > 0$ п
$y_i(x^0) < 0$. Решим сначала задачу

$$\max\{-\xi_1 \mid a_i^T x - \rho_i \xi_1 \leqslant b_i,\ i \in I_L,\ 0 \leqslant x \leqslant c;\ \xi_1 \geqslant 0\}. \quad (7.2$$

[1]) См. примечание 2 на стр. 10. — *Прим. ред.*

Это — задача линейного программирования, и ее можно решить симплексным методом или другим методом возможных направлений. Исходным возможным решением этой задачи будет $x = x^0$, $\xi_1 = \max\left\{-\frac{y_i(x^0)}{\rho_i} \middle| \rho_i > 0\right\}$. Методы возможных направлений конечны в линейном случае (см. гл. 9), поэтому, если множество R непусто, после конечного числа шагов мы получим $\xi_1 = 0$. Точка x_1^0, являющаяся решением (7.2.1), удовлетворяет всем линейным ограничениям (7.1.1). Если $y_i(x_1^0) \geqslant 0$ для $i \in I_C$, то $x_1^0 \in R$, и мы построили исходное возможное решение. Если $y_i(x_1^0) < 0$ для некоторых $i \in I_C$, введем дополнительную переменную ξ_2 и неотрицательные числа ρ_i, $i \in I_C$, такие, что $\rho_i = 0$ при $y_i(x_1^0) \geqslant 0$ и $\rho_i > 0$ при $y_i(x_1^0) < 0$. Решим задачу

$$\max\{-\xi_2 \mid f_i(x) - \rho_i \xi_2 \leqslant b_i,\ i \in I_C;\ a_i^T x \leqslant b_i,$$
$$i \in I_L;\ 0 \leqslant x \leqslant c;\ \xi_2 \geqslant 0\}. \quad (7.2.2)$$

Исходное возможное решение задачи (7.2.2) легко получить: $x = x_1^0$, $\xi_2 = \max\left\{-\frac{y_i(x_1^0)}{\rho_i} \middle| \rho_i > 0\right\}$. Поскольку (7.2.2) — задача выпуклого программирования типа (7.1.1), для ее решения можно использовать те же методы, что и для (7.1.1).

Теорема 1. *Если R непусто и удовлетворяет* C1, *то, применяя для решения* (7.2.2) *любой метод возможных направлений, получим $\xi_2 = 0$ после конечного числа шагов.*

Доказательство. Из условия C1 следует существование вектора x, удовлетворяющего требованиям $f_i(x) < b_i$, $i \in I_C$; $a_i^T x \leqslant b_i$, $i \in I_L$, $0 \leqslant x \leqslant c$. Поэтому если в (7.2.2) не требовать $\xi_2 \geqslant 0$, то последовательность решений будет сходиться к значению $\xi_2 < 0$, и после конечного числа итераций ξ_2 становится неположительным (п. 7.1, свойство 3 метода возможных направлений). Следовательно, при условии $\xi_2 \geqslant 0$ значение $\xi_2 = 0$ достигается после конечного числа итераций. Теорема доказана.

Вместо того, чтобы сначала решать задачу (7.2.1), а затем — (7.2.2), можно решать одну задачу

$$\max\{-\xi \mid f_i(x)-\rho_i\xi \leqslant b_i,\ i\in I_C;\ a_i^Tx-\rho_i\xi\leqslant b_i,\ i\in I_L;\ 0\leqslant x\leqslant c;\ \xi\geqslant 0\}, \quad (7.2.3)$$

где $\rho_i \geqslant 0$ и положительно тогда и только тогда, когда $y_i(x^0)<0$. Исходным возможным решением этой задачи является $x=x^0$, $\xi=\max_i\left\{-\frac{y_i(x^0)}{\rho_i}\,\middle|\,\rho_i>0\right\}$, так что можно сразу применить метод возможных направлений. Но, в то время как задачи (7.2.1) и (7.2.2) решаются за конечное число шагов, в случае (7.2.3) это не обязательно.

Можно также решать следующую задачу:

$$\max\{F(x)-M\xi \mid f_i(x)-\rho_i\xi\leqslant b_i,\ i\in I_C;\ a_i^Tx-\rho_i\xi\leqslant b_i,\ i\in I_L;\ 0\leqslant x\leqslant c;\ \xi\geqslant 0\}, \quad (7.2.4)$$

где M — достаточно большое положительное число.

Теорема 2. *Если задача* (7.1.1) *имеет конечный максимум* x', $F(x')<\infty$, *то существует такое число* M_1, *что при* $M\geqslant M_1$ x_1 *является также оптимальным решением задачи* (7.2.4).

Доказательство. Так как x' — максимальное решение (7.1.1), то в силу результатов п. 2.6 [теорема 6 и (2.6.6)] существуют такие неотрицательные числа u_i', v_{1j}' и v_{2j}', что

$$g(x')=\sum u_i'q_i(x')+\sum u_i'a_i-\sum v_{1j}'e_j+\sum v_{2j}'e_j,$$
$$u'^Ty'=0,\quad v_1'^Tx'=0,\quad v_2'^T(c-x')=0.$$

С другой стороны, если x, y, ξ — возможное решение (7.2.4) и система

$$g(x)=\sum u_iq_i(x)+\sum u_ia_i-\sum v_{1j}e_j+\sum v_{2j}e_j,$$
$$-M=-\sum u_i\rho_i-v_\xi,$$
$$u^Ty=0,\quad v_1^Tx=0,\quad v_2^T(c-x)=0,\quad \xi v_\xi=0,$$
$$u\geqslant 0,\quad v_1\geqslant 0,\quad v_2\geqslant 0,\quad v_\xi\geqslant 0$$

совместна, то x, y, ξ — оптимальное решение (7.2.4). В предположении $M\geqslant\sum u_i'\rho_i$ решением этой системы будет

$$x=x',\quad y=y',\quad \xi=0,\quad u=u',\quad v_1=v_1',$$
$$v_2=v_2'\quad\text{и}\quad v_\xi=M-\sum u_i'\rho_i.$$

Таким образом, положив $M_1=\sum u_i'\rho_i$, завершаем доказательство теоремы.

Если $F(x)$ — нелинейная функция, предпочтительнее двухэтапное решение, так как на первой фазе оптимизируемая функция линейна и не требуется многократных вычислений градиента. Задачу типа (7.2.4) приходится решать, например, когда в результате накопления ошибок округления в процессе вычислений мы получим точку x^k, лежащую вне области R, но поблизости от нее.

До сих пор ничего не сказано о выборе ρ_i. Можно, например, положить $\rho_i=0$ при $y_i(x^0)\geqslant 0$ и $\rho_i=1$ при $y_i(x^0)<0$. Или, например, $\rho_i=0$ при $y_i(x^0)\geqslant 0$ и $\rho_i=-y_i(x^0)$ при $y_i(x^0)<0$. В последнем случае $x=x^0$, $\xi=1$ — возможное решение измененной задачи и для него выполняются как равенства все условия, нарушаемые для точки x^0 в исходной задаче. Это можно использовать при корректировании ошибок округления. [Если выбрать $\rho_i=1$ при $y_i(x^0)<0$, проверяемое решение может не попасть на те граничные поверхности, которым оно, по-видимому, принадлежало бы при отсутствии ошибок округления.]

7.3. Определение подходящего возможного направления.

Пусть для произвольного $x\in R$ множества индексов $I_C(x)\subset I_C$, $I_L(x)\subset I_L$, $J^-(x)\subset J$, $J^+(x)\subset J$ определены следующим образом:

$i\in I_C(x)$ тогда и только тогда, когда $f_i(x)=b_i$, (7.3.1)

$i\in I_L(x)$ тогда и только тогда, когда $a_i^Tx=b_i$, (7.3.2)

$j\in J^-(x)$ тогда и только тогда, когда $x_j=0$, (7.3.3)

$j\in J^+(x)$ тогда и только тогда, когда $x_j=c_j$. (7.3.4)

Положим далее

$$S(x)=\{s\,|\,q_i(x)^T s\leqslant 0,\ i\in I_C(x);\ a_i^T s\leqslant 0,\ i\in I_L(x);$$
$$s_j\geqslant 0,\ j\in J^-(x);\ s_j\leqslant 0,\ j\in J^+(x)\}. \quad (7.3.5)$$

Введем искусственную переменную σ, положительны числа Θ_i и определим

$$S'(x)=\{(s,\sigma)\,|\,q_i(x)^T s+\Theta_i\sigma\leqslant 0,\ i\in I_C(x);\ a_i^T s\leqslant 0,$$
$$i\in I_L(x);\ s_j\geqslant 0,\ j\in J^-(x);\ s_j\leqslant 0,\ j\in J^+(x)\}. \quad (7.3.6)$$

Рассмотрим теперь вектор (s,σ), удовлетворяющий тре бованиям:

1) $(s,\sigma)\in S'(x)$, (7.3.7)

2) $-g(x)^T s+\sigma\leqslant 0$, (7.3.8)

3) требование нормализации, (7.3.9)

4) значение σ максимизируется. (7.3.10)

Задачу определения вектора (s,σ), удовлетворяющего эти требованиям, будем называть задачей *выбора направлен* Произвольный вектор (s,σ), удовлетворяющий (7.3.7) пр $\sigma>0$, определяет подходящее возможное направление, по скольку $q_i(x)^T s<0$ при $i\in I_C(x)$ и $g(x)^T s>0$. Нормали зация избавляет нас от неограниченных решений задачи. Нор мализация должна быть такой, что если (s,σ) удовлетворя условиям (7.3.7) и (7.3.8), то существует такое $\bar\beta>0$, чт для $0\leqslant\beta\leqslant\bar\beta$ вектор $(\beta s,\beta\sigma)$ удовлетворяет (7.3.7)— (7.3.9), и если s_1 и s_2 — нормализованные векторы, вектор $\lambda s_1+(1-\lambda)s_2$ также нормализован при любом $0\leqslant\lambda\leqslant 1$.

Если в оптимальном решении (s',σ') задачи (7.3.7)—(7.3.10) значение σ' равно нулю, то система линейных неравенс (7.3.7), (7.3.8) и $\sigma>0$ несовместна и в силу теоремы 3 п. 2 существуют такие неотрицательные числа u_i, v_j^-, v_j^+ и

что

$$\sum_{i\in I_C(x)}u_iq_i(x)+\sum_{i\in I_L(x)}u_ia_i-\sum_{j\in J^-(x)}v_j^-e_j+$$
$$+\sum_{j\in J^+(x)}v_j^+e_j-u_0g(x)=0, \quad (7.3.11)$$

$$\sum_{i\in I_C(x)}u_i\Theta_i+u_0=1. \quad (7.3.12)$$

Если $u_0>0$, то из (7.3.11) и теоремы 6 п. 2.6 следует, что x является максимумом $F(x)$ на R.

Если $u_0=0$, то в силу (7.3.12) $u_i>0$ по крайней мере для одного $i\in I_C(x)$, и из (7.3.11) следует, что условие C1 не удовлетворяется[1]). Таким образом, при выполнении условия C1 x — точка максимума $F(x)$ на R, если для всех (s,σ), удовлетворяющих (7.3.7) и (7.3.8), $\sigma\leqslant 0$. Нами доказана

Теорема 1. *При соблюдении условия* C1 *точка* $x\in R$ *является максимумом* $F(x)$ *на* R *тогда и только тогда, когда* $\sigma\leqslant 0$ *для всех* $(s,\sigma)\in S'(x)$, *удовлетворяющих условию* $-g(x)^T s+\sigma\leqslant 0$.

Выбор Θ_i может быть произвольным. Можно, например, положить $\Theta_i=1$ для всех $i\in I_C(x)$, но можно и увеличить значение Θ_i, если мы неоднократно возвращаемся на одну и ту же гиперповерхность.

В случае $I_C=0$, т. е. когда нелинейные ограничения отсутствуют, задача выбора направления сводится к следующей:

1) $s\in S(x)$, (7.3.13)

2) требование нормализации, (7.3.14)

3) значение $g(x)^T s$ максимизируется. (7.3.15)

Произвольное s, удовлетворяющее (7.3.13) и $g(x)^T s>0$, будет подходящим возможным направлением. Если в максимальном решении задачи выбора направления $g(x)^T s'=0$, то x — точка максимума $F(x)$ на R.

Требование (7.3.10) или (7.3.15) ведет к выбору лучшего подходящего возможного направления среди нормализованных направлений. Интересно, что многие методы решения задач линейного, квадратичного и выпуклого программирования,

[1]) См. теорему 2 п. 2.6. — *Прим. ред.*

совершенно различные с первого взгляда, отличаются лишь способами нормализации, определяющими выбор направления s, и программами вычислений, зависящими от этого выбора.

Приведем примеры возможных способов нормализации:

N1. $s^T s \leqslant 1$;

N2. $-1 \leqslant s_j \leqslant 1$ для всех j;

N3. $s_j \leqslant 1$ при $g_j(x) > 0$,
$s_j \geqslant -1$ при $g_j(x) < 0$;

N4. $\sigma \leqslant 1$ в (7.3.7) — (7.3.10),
$g(x)^T s \leqslant 1$ в (7.3.13) — (7.3.15);

N5. $q_i(x)^T s + \Theta_i \sigma \leqslant y_i(x)$, $i \in I_C$,
$a_i^T s \leqslant y_i(x)$, $i \in I_L$,
$-x_j \leqslant s_j \leqslant c_j - x_j$, $j \in J$.

Для сходимости итерационного процесса необходимо, чтобы направления s^k были ограничены. Это всегда справедливо при нормализациях N1 и N2. В других случаях можно, например, требовать, чтобы абсолютно наибольшая компонента s^k не превосходила заданного числа. Если же она превосходит это число, можно рассматривать вектор αs^k, где $\alpha < 1$ выбирается так, чтобы абсолютно наибольшая компонента лежала в заданных пределах.

Нормализации N1 — N5 изучаются в гл. 8.

Нормализация N5 отличается от других учетом и тех ограничений, которым текущее решение x не удовлетворяет как равенствам.

Фиксируем некоторую точку $\bar{x} \in R$ и положим $s = x - \bar{x}$ ($\bar{x}$ — фиксированная точка, x — переменная), $\sigma = x_0$. Тогда соотношения (7.3.7), (7.3.8), N5 и (7.3.10) приводят к задаче

$$\max \{x_0 \mid \bar{q}_i^T x + \Theta_i x_0 \leqslant \bar{q}_i^T \bar{x} + b_i - f_i(\bar{x}),\ i \in I_C;\ a_i^T x \leqslant b_i,\ i \in I_L;\ 0 \leqslant x_j \leqslant c_j,\ j \in J;\ -\bar{g}^T x + x_0 \leqslant -\bar{g}^T \bar{x}\}, \quad (7.3.16)$$

где $\bar{q}_i = q_i(\bar{x})$ и $\bar{g} = g(\bar{x})$. Так как задача (7.3.16) может иметь неограниченное решение, то нормализация N5 не является нормализацией в полном смысле слова. Если решение (7.3.16) не ограничено, подходящий вектор s можно построить, зная экстремальный луч, легко получаемый из результатов решения (7.3.16) [см. п. 3.2 после формулы (3.2.5)].

7.4. Определение длины шага.

После выбора в точке x подходящего возможного направления s будем улучшать решение, двигаясь в направлении s.

Определяя

$$\lambda' = \max \{\lambda \mid x + \lambda s \in R\}, \quad (7.4.1)$$

получим, что длина шага λ не должна превосходить λ' (если некоторые c_j равны бесконечности, λ' может быть равно бесконечности). Пусть λ'' — наименьшее λ, такое, что

$$F(x + \lambda'' s) = \max_{\lambda} F(x + \lambda s), \quad (7.4.2)$$

т. е. $\lambda'' = \max \{\lambda \mid F(x + \mu s) < F(x + \lambda s)$ для всех $\mu < \lambda\}$. Тогда $0 < \lambda'' \leqslant \infty$. Положим

$$\lambda = \min(\lambda', \lambda''), \quad (7.4.3)$$

т. е.

$$F(x + \lambda s) = \max \{F(x + \mu s) \mid \mu \leqslant \lambda'\}. \quad (7.4.4)$$

Если $\lambda = \infty$, то из условия регулярности С3 следует, что $m = \infty$ и конечного максимума не существует. Действительно, из условия $m < \infty$ следует ограниченность $\{y \in R \mid F(y) \geqslant F(x)\}$, так что $\lambda = \infty$ не может иметь места.

Задача (7.4.4) одномерная. Ее легко решить, если все ограничения линейны, а оптимизируемая функция линейная или квадратичная. Задачи определения λ' и λ'' в общем случае аналогичны. В (7.4.1) необходимо для каждого i определить наибольший корень уравнения $f_i(x + \lambda s) = b_i$ и затем выбрать наименьший из них. Для всякого $i \in I_C$ это можно сделать, например, методом Ньютона (заметим, что λ — единственная переменная в этой задаче, а x и s — известные векторы). В случае линейных ограничений λ' просто определяется из условий

$$\lambda' = \min(\lambda_1', \lambda_2', \lambda_3'), \quad (7.4.5)$$

причем

$$\lambda_1' = \min_i \left\{ \frac{y_i(x)}{-t_i} \,\middle|\, t_i < 0,\ i \in I_L - I_L(x) \right\}, \quad (7.4.6)$$

где

$$t_l = -a_l^T s; \qquad (7.4.7)$$

$$\lambda_2' = \min_j \left\{ \frac{x_j}{-s_j} \,\middle|\, s_j < 0,\ j \in J - J^-(x) \right\}, \qquad (7.4.8)$$

$$\lambda_3' = \min_j \left\{ \frac{c_j - x_j}{s_j} \,\middle|\, s_j > 0,\ j \in J - J^+(x) \right\}. \qquad (7.4.9)$$

Далее, определим λ следующим образом: если $g(x + \lambda' s)^T s \geqslant 0$, то $\lambda = \lambda'$ [$F(x)$ — вогнутая функция]; если $g(x + \lambda' s)^T s < 0$, искусственным приемом определим λ'', такое, что $g(x + \lambda'' s)^T s = 0$ (λ — единственная переменная, x и s — известные векторы).

7.5. Алгоритмы.

В этом пункте мы рассмотрим алгоритмы решения, основанные на различных нормализациях. Алгоритм P1 основан на одной из нормализаций N1 — N4; алгоритм P2 основан на нормализации N5. Для обеспечения сходимости $F(x^k)$ к $m = \max\{F(x) \mid x \in R\}$ в P1 необходимо использовать специальные приемы, так называемые меры предосторожности против „зигзагообразного" движения, так как иначе можно построить пример, в котором $\lim F(x^k) < m$. Опишем некоторые приемы [1]), обеспечивающие сходимость.

AZ1.

a. Пусть $\varepsilon > 0$ — произвольное число.

b. Для $x \in R$ определим

$$I_C(x, \varepsilon) = \{i \in I_C \mid b_i - \varepsilon \leqslant f_i(x) \leqslant b_i\}, \qquad (7.5.1)$$

$$I_L(x, \varepsilon) = \{i \in I_L \mid b_i - \varepsilon \leqslant a_i^T x \leqslant b_i\}, \qquad (7.5.2)$$

$$J^-(x, \varepsilon) = \{j \in J \mid 0 \leqslant x_j \leqslant \varepsilon\}, \qquad (7.5.3)$$

$$J^+(x, \varepsilon) = \{j \in J \mid c_j - \varepsilon \leqslant x_j \leqslant c_j\}, \qquad (7.5.4)$$

$$H(x, \varepsilon) = I_C(x, \varepsilon) + I_L(x, \varepsilon) + J^-(x, \varepsilon) + J^+(x, \varepsilon), \qquad (7.5.5)$$

$$H(x) = H(x, 0). \qquad (7.5.6)$$

[1]) В дальнейшем эти приемы обозначаются AZ (anti-zigzagging precaution). — *Прим. перев.*

Ясно, что при $\varepsilon = 0$

$$I_C(x, 0) = I_C(x), \quad I_L(x, 0) = I_L(x),$$

$$J^-(x, 0) = J^-(x) \text{ и } J^+(x, 0) = J^+(x).$$

c. Положим

$$S'(x, \varepsilon) = \{(s, \sigma) \mid q_i(x)^T s + \Theta_l \sigma \leqslant 0,\ l \in I_C(x, \varepsilon);\ a_l^T s \leqslant 0,\ l \in I_L(x, \varepsilon);\ s_j \geqslant 0,\ j \in J^-(x, \varepsilon);\ s_j \leqslant 0,\ j \in J^+(x, \varepsilon)\}. \qquad (7.5.7)$$

d. Заменим $S'(x)$ в условии (7.3.7) задачи выбора направления на $S(x, \varepsilon)$.

e. Если в ограниченном оптимальном решении задачи выбора направления значение $\sigma_{опт}$ меньше ε, заменим ε на $\varepsilon/2$, $S'(x, \varepsilon)$ на $S'(x, \varepsilon/2)$ и снова решим задачу выбора направления. [Если $\sigma_{опт} = 0$ и $S'(x, \varepsilon) = S'(x)$, то x — максимум $F(x)$ на R.]

AZ2.

a. Пусть $\varepsilon > 0$ — произвольное число.

b. Пусть определено x^k, т. е. решена k-я задача выбора направления. Пусть $h \in H(x^k)$ такое, что $h \notin H(x^{k-1})$, но $h \in H(x^l)$ для некоторого $l \leqslant k - 2$. Для такого значения h будем сохранять требование $n_h^T s + \Theta_h \sigma \leqslant 0$ во всех задачах выбора направления (7.3.7) — (7.3.10) либо (7.3.13) — (7.3.15), следующих за k-й до тех пор, пока не встретится случай c. Таким образом, даже при $h \notin H(x^{k+1})$ мы требуем $n_h^T s + \Theta_h \sigma \leqslant 0$ в $(k+1)$-й задаче выбора направления. (n_h — внешняя нормаль в точке x^k к h-й гиперповерхности, т. е. n_h равно $q_i(x^k)$, a_l, $-e_j$ либо e_j; $\Theta_h = 0$, если $h \notin I_C$.)

c. Если $\lambda_k > \lambda_k'$, дополнительные требования опускаются в следующей задаче выбора направления; они опускаются в текущей задаче, если $\sigma_{опт} < \varepsilon$ для выбранного $s_{опт}$. Таким образом, k-я задача снова решается, но без дополнительных требований, причем в новой задаче ε заменяется на $\varepsilon/2$.

В AZ1 принимаются во внимание те гиперповерхности, для которых переменные $y_i(x)$, x_j или $c_j - x_j$ хотя и не равны нулю, но не превосходят ε. Этим мы предотвращаем очень малые шаги (ε в процессе решения стремится к нулю, поэтому оно не мешает нам в окрестности оптимальной точки).

В AZ2, если мы вторично попали на некоторую гиперповерхность, для того, чтобы не попасть на нее в третий раз, вводятся дополнительные требования. Дополнительное требование $n_h^T s + \Theta_h \sigma \leq 0$ может сохраняться, даже когда мы находимся довольно далеко от соответствующей гиперповерхности. Для уменьшения этой опасности дополнительные требования опускаются, когда $\lambda_k < \lambda_k'$. Прием AZ1 простейший, однако, по-видимому, AZ2 более эффективный, так как при его использовании мы скорее покинем некоторую область, если она не содержит оптимального решения.

В дальнейшем нам понадобится еще один прием, который мы сформулируем для случая линейных ограничений.

AZ3.

a. Потребуем $n_h^T s = 0$ вместо $n_h^T s \leq 0$, если мы вторично попали на соответствующую граничную гиперповерхность. Это требование сохраняется до тех пор, пока не встретится случай b.

b. В некоторой задаче выбора направления $g(x^k)^T s_{опт} = 0$. В этом случае по крайней мере одно из требований $n_h^T s = 0$ заменяем на $n_h^T s \leq 0$.

Прием AZ3 применяется в квадратичном программировании, описанном в гл. 10; его основное назначение — сделать метод конечным. В гл. 11 мы покажем, как можно использовать этот прием для ускорения сходимости в случае общей задачи выпуклого программирования. Для этого AZ3 применяется совместно с AZ1.

Итак, получены два следующих алгоритма решения задачи выпуклого программирования.

Алгоритм P1.

1. Отправляемся от точки $x^0 \in R$. Если такая точка неизвестна, применяем методику, описанную в п. 7.2. Возьмем произвольное малое $\varepsilon > 0$ и очень малое $\varepsilon' > 0$.

2. Предположим, что $x^0, x^1, \ldots, x^k$ определены. Решим k-ю задачу выбора направления:

a) $(s, \sigma) \in S'(x^k, \varepsilon)$ либо $(s, \sigma) \in S'(x^k)$ плюс дополнительные требования AZ2;

b) нормализация N1, N2, N3 или N4;

c) $-g(x^k)^T s + \sigma < 0$;

d) значение σ максимизируется.

3. Если в этой задаче $\sigma_k' < \varepsilon$, заменим ε на $\varepsilon/2$, $S'(x^k, \varepsilon)$ на $S'(x^k, \varepsilon/2)$ в AZ1, опустим дополнительные требования при AZ2 и снова решим задачу выбора направления.

Если $\sigma_k' = 0$ и $S'(x^k, \varepsilon) = S'(x^k)$ в AZ1 либо нет дополнительных требований при AZ2, задача решена [в предположении, что R удовлетворяет C1; в противном случае добавляем к b_i малые величины Δb_i для тех i, $i \in I_C$, для которых в (7.3.11) $u_i > 0$; см. п. 2.6]. Если $\sigma_k' \geq \varepsilon$, переходим к шагу 4.

4. Определим λ_k, используя (7.4.1) и (7.4.4). Если $\lambda_k = \infty$, то из условия C3 следует, что $F(x)$ не ограничена на R. Если $\lambda_k < \lambda_k'$ и используется AZ2, в следующей задаче выбора направления опускаются дополнительные требования.

5. Определим

$$x^{k+1} = x^k + \lambda_k s^k \tag{7.5.8}$$

и $F(x^{k+1})$.

6. Если $F(x^{k+1}) - F(x^k) < \varepsilon'$ и k-я задача выбора направления не содержала дополнительных требований AZ1, AZ2, процесс решения прекращается. Для проверки близости к оптимальному решению можно решить задачу линейного программирования (2.6.7) и получить оценку сверху возможного приращения $F(x)$. Если она достаточно мала, точка x^k близка к оптимальному решению. Во всех остальных случаях повторяем шаги 2—5.

Алгоритм P2.

1. Отправляемся от точки $x^0 \in R$. Пусть ε' — заданное малое положительное число.

2. Предположим, что $x^0, x^1, \ldots, x^k$ определены.

Решим вспомогательную задачу линейного программирования

$$\max \{x_0 \mid q_i(x^k)^T x + \Theta_i x_0 \leq q_i(x^k)^T x^k + b_i - f_i(x^k),$$
$$i \in I_C;\ a_i^T x \leq b_i,\ i \in I_L;\ 0 \leq x_j \leq c_j,\ j \in J;$$
$$-g(x^k)^T x + x_0 \leq -g(x^k)^T x^k\}.$$

3. Если $x_{0\max} = 0$, то x^k — оптимальное решение.

Если $0 < x_{0\max} < \infty$, положим $s^k = x_{\max} - x^k$.

Если $x_{0\max} = \infty$, построим s^k с помощью экстремального луча, лежащего в допустимой области и ведущего к неограниченному решению задачи линейного программирования.

4. С помощью (7.4.1) и (7.4.4) определим λ_k. Если $\lambda_k=\infty$, то из условия С3 следует, что функция $F(x)$ не ограничена на R.

5. Определим

$$x^{k+1}=x^k+\lambda_k s^k \tag{7.5.8}$$

и $F(x^{k+1})$.

6. Если $F(x^{k+1})-F(x^k)<\varepsilon'$, процесс решения прекращается.

Решением задачи (2.6.7) можно определить, близко ли x^{k+1} к оптимальному решению или нет. Если $F(x^{k+1})-F(x^k)\geqslant\varepsilon'$, повторяем шаги 2—5.

Заметим, что решение линеаризованной задачи (2.6.7) может быть неограниченным или очень большим, хотя точка x^{k+1} близка к оптимальному решению. В гл. 11 показано, как можно улучшить проверку. Там также описаны некоторые новые алгоритмы, полученные из P1 и P2. Эти алгоритмы порождают, как правило, более быстро сходящуюся последовательность решений.

7.6. Сходимость последовательности решений.

Докажем сначала три леммы.

Лемма 1. *Если $x\in R$, то либо в точке x достигается максимум $F(x)$ на R, либо существует число $\delta>0$ и вектор $s\in S(x)$ такие, что $q_i(x)^T s\leqslant-\Theta_i\delta$ для всех $i\in I_C(x)$ и $g(x)^T s\geqslant\delta$.*

Доказательство. Предположим, что x не является точкой максимума $F(x)$ на R. Из теоремы 1 п. 7.3 известно, что в этом случае в оптимальном решении задачи выбора направления $\sigma_{\max}>0$. Следовательно, можно положить $\delta=\sigma_{\max}$ и $s=s_{\max}$.

Лемма 2. *Пусть $F(x)$ — произвольная функция, определенная на замкнутом выпуклом множестве R. Ее градиент $g(x)$ — непрерывная вектор-функция. Пусть $x^k\in R$, $k=0, 1, 2, \ldots,$ — ограниченная последовательность точек; s^k, $k=0, 1, 2, \ldots,$ — ограниченная последовательность векторов и $\lambda_k>0$ — последовательность скаляров, причем*

для всех k справедливо $x^{k+1}=x^k+\lambda_k s^k$ и $g(x^k+\lambda s^k)^T s^k>0$ при $0\leqslant\lambda<\lambda_k$. Предположим, что для некоторого $\delta>0$ и подпоследовательностей $y^l=x^{k_l}$, $t^l=s^{k_l}$, $\mu_l=\lambda_{k_l}$ выполняется условие $g(y^l)^T t^l\geqslant\delta$ для всех l. Тогда $\sum\mu_l$ сходится.

Доказательство. Определим $\bar{\mu}_l=\max\left\{\mu\leqslant\mu_l \mid g(y^l+\mu t^l)^T t^l\geqslant\frac{\delta}{2}\right\}$. Тогда для любого $p>0$

$$F(x^{k_p+1})-F(x^0)=\sum_{h=0}^{k_p}\{F(x^{h+1})-F(x^h)\}\geqslant$$
$$\geqslant\sum_{l=0}^{p}\{F(x^{k_l+1})-F(y^l)\}\geqslant$$
$$\geqslant\sum_{l=0}^{p}\{F(y^l+\bar{\mu}_l t^l)-F(y^l)\}\geqslant\frac{\delta}{2}\sum_{l=0}^{p}\bar{\mu}_l$$

и из ограниченности $F(x)$ следует, что $\sum_{l=1}^{\infty}\bar{\mu}_l$ сходится. Если $\bar{\mu}_l<\mu_l$, то $g(y^l+\bar{\mu}_l t^l)^T t^l=\frac{\delta}{2}$. Если $\sum_{l=1}^{\infty}\mu_l$ не ограничена, этот случай встречается бесконечное число раз. Будем рассматривать лишь соответствующие значения l ($l\in L_1$). Последовательность y^l ограничена, следовательно, существует предельная точка y'. Отсюда вытекает, что для некоторой подпоследовательности точек y^l ($l\in L_2\subset L_1$) $\bar{\mu}_l<\mu_l$ и $\lim y^l=y'$. Значит существуют такие числа δ_1, δ_2 и l_1, что $\frac{\delta}{2}<\delta_1<\delta_2<\delta$, и для $l\in L_2$ и $l\geqslant l_1$ имеют место неравенства $g(y')^T t^l\geqslant\delta_2$ и $g(y'+\bar{\mu}_l t^l)^T t^l\leqslant\delta_1$. Последовательность t^l, $l\in L_2$, также ограничена. Следовательно, некоторая ее подпоследовательность сходится к вектору t'. Пусть это будет последовательность $l\in L_3\subset L_2$ (для конечного или бесконечного множества l возможно $t^l=t'$). Тогда существуют такие числа δ_3, δ_4 и $l_2\geqslant l_1$, что $\delta_1<\delta_3<\delta_4<\delta_2$, и

для $l \in L_3$ и $l \geqslant l_2$ справедливы неравенства $g(y')^T t' \geqslant \delta_4$ и $g(y' + \mu_l t')^T t' \leqslant \delta_3$. Но в силу непрерывности $g(x)$ и сходимости $\sum_{l=1}^{\infty} \bar{\mu}_l$ это невозможно. Следовательно, $\sum_{l=1}^{\infty} \mu_l$ сходится, что доказывает лемму.

Следствие. Если s^k $(k = 0, 1, 2, \ldots)$ — ограниченная последовательность подходящих векторов, $x^k \in R$ — ограниченная последовательность точек и для некоторого $\delta > 0$ и всех k выполняется $g(x^k)^T s^k \geqslant \delta$, то $\lambda_k = \lambda_k'' < \lambda_k'$ может встретиться лишь конечное число раз.

Доказательство. Поскольку выполнены условия леммы 2, $\sum_{k=1}^{\infty} \lambda_k$ сходится, так что для всех достаточно больших k $\lambda_k = \bar{\lambda}_k$ ($\bar{\lambda}_k$ соответствует $\bar{\mu}$ в доказательстве леммы 2), следовательно, $g(x^k + \lambda_k s^k)^T s^k \geqslant \delta/2 > 0$ при достаточно больших k, и $\lambda_k = \lambda_k'' < \lambda_k'$ не может иметь места.

Лемма 3. *Пусть $F(x)$ — произвольная функция, градиент $g(x)$ которой непрерывная вектор-функция, $\{x^k | k \in K\}$ — конечное или бесконечное ограниченное множество точек. Пусть, далее, найдется такое число $\delta > 0$, что для каждого $k \in K$ существует ограниченный вектор $s^k = s(x^k)$, удовлетворяющий условию $g(x^k)^T s^k \geqslant \delta$. Если для $k \in K$ $\lambda_k = \max\{\lambda | F(x^k + \mu s^k) > F(x^k)$ для всех $\mu < \lambda\}$, то $\inf_k \lambda_k > 0$.*

Доказательство. Из определения следует, что λ_k либо бесконечно, либо равно наименьшему значению λ, для которого $F(x^k + \lambda s^k) = F(x^k)$. Пусть $\mu_k = \max\left\{\mu | g(x^k + \mu s^k)^T s^k \geqslant \frac{\delta}{2}\right.$ для всех $\left.\mu \leqslant \mu_k\right\}$. Тогда $F(x^k + \mu_k s^k) - F(x^k) \geqslant \mu_k \frac{\delta}{2}$, так что $\lambda_k \geqslant \mu_k$. Предположим, что существует такая последовательность точек, что μ_k стремится к нулю. Для этой последовательности при любом k будет $g(x^k)^T s^k \geqslant \delta$ и $g(x^k + \mu_k s^k)^T s^k = \frac{\delta}{2}$. Так же как и при доказательстве леммы 2, можно показать, что это невозможно. Следовательно, $\inf_k \mu_k > 0$, а значит, и $\inf_k \lambda_k > 0$.

Следствие. Если $f_i(x^k) \leqslant b_i$, $q_i(x^k)^T s^k < -\Theta_i \bar{\delta}$ для всех k, s^k ограничены $x^{k+1} = x^k + \lambda_k s^k$ и $\sum_{1}^{\infty} \lambda^k$ сходится, то существует такое k_1, что $f_i(x^k) < b_i$ для $k \geqslant k_1$.

Доказательство. Положим $F(x) = -f_i(x)$, $g(x) = -q_i(x)$, $\delta = \Theta_i \bar{\delta}$ и применим лемму 3.

Докажем теперь следующую теорему.

Теорема 1. *Алгоритм P1, использующий в задаче выбора направления одну из нормализаций N1 — N4, а также AZ1 или AZ2, при $\varepsilon' = 0$ порождает последовательность точек x^k такую, что $\lim_{k \to \infty} F(x^k) = m = \max\{F(x) | x \in R\}$ в предположении, что R и $F(x)$*[1] *удовлетворяют условиям C1 и C3 соответственно.*

Доказательство. Если общее число шагов конечно и оптимальное решение ограничено, теорема, очевидно, справедлива. Если в некоторой точке получено бесконечное решение ($\lambda_k = \infty$ для некоторого k), то $m = \infty$ в силу условия C3 и теоремы 1 п. 7.1.

Предположим теперь, что число шагов не ограничено. Если последовательность точек x^k не ограничена, из условия C3 следует, что $\lim F(x^k) = \infty$, и теорема справедлива. Предположим поэтому, что последовательность x^k, так же как и последовательность $F(x^k)$, ограничена. Пусть (s^k, σ^k) — ограниченная последовательность векторов, коллинеарных оптимальным решениям соответствующих задач выбора направления (если необходимо, включающих дополнительные требования приемов AZ, обеспечивающих сходимость). Если $g(x^k)^T s^k \geqslant \delta$ для некоторого $\delta > 0$ и всех k (исключая, быть может, конечное число) и $\sigma^k \geqslant \delta$, а значит, и

[1] Имеются в виду выпуклое множество R и вогнутая функция $F(x)$. — *Прим. перев.*

$x^{k+1}=x^k+\lambda_k s^k$, то $\sum\limits_{k=1}^{\infty}\lambda_k$ сходится (лемма 2). Положим $\bar x=\lim x^k$ и для $\varepsilon'>0$ определим множество $R(\bar x, \bar\varepsilon)=\{x\in R \mid |x_j-\bar x_j|\leqslant\bar\varepsilon$ для всех j; $|y_i(x)-y_i(\bar x)|\leqslant\bar\varepsilon$ для всех $i\}$. Выберем $\bar\varepsilon$ столь малым, что для любого $x\in R(\bar x, \bar\varepsilon)$ справедливо $H(x)\subset H(\bar x)$, иначе говоря, не существует $x\in R(\bar x, \bar\varepsilon)$, принадлежащего некоторой граничной гиперповерхности, которой не принадлежит $\bar x$. Соотношение $\sigma^k\geqslant\delta$ означает, что после конечного числа шагов величина ε в AZ1 или AZ2 перестанет убывать. Выберем $k(\bar\varepsilon)$ столь большим, что для $k\geqslant k(\bar\varepsilon)$ ε не убывает, $x^k\in R(\bar x, \bar\varepsilon)$ и $\lambda_k=\lambda_k''<\lambda_k'$ не имеет места (это возможно согласно следствию леммы 2). При использовании AZ1 должно также выполняться условие $\bar\varepsilon<\varepsilon$ (имеется в виду предельное значение ε в AZ1). Тогда для $k\geqslant k(\bar\varepsilon)$ выполняется условие $H(x^k, \varepsilon)\supset H(\bar x)$, и мы можем применить следствие леммы 3, показывающее, что $f_i(x^k)<b_i$ для всех $i\in I_C(\bar x)\subset I_C(x^k, \varepsilon)$ при достаточно большом k. Следовательно, мы не можем попасть на эти нелинейные граничные гиперповерхности. Мы также не можем благодаря AZ1 вернуться на граничную гиперплоскость, если мы покинули ее. Кроме того, не может быть $\lambda_k<\lambda_k'$. Таким образом, мы пришли к противоречию. При AZ2 и $k\geqslant k(\bar\varepsilon)$, достаточно большом, мы не можем вернуться на граничную гиперплоскость, если только мы были на ней дважды, так что у нас снова не будет выбора после конечного числа шагов, т. е. мы придем к тому же противоречию. Следовательно, σ^k становится меньше произвольно малой величины, и мы систематически заменяем ε на $\varepsilon/2$. Пусть $y^l=x^{k_l}$ — подпоследовательность точек x^k, такая, что при каждом l ε заменяется на $\varepsilon/2$, и пусть $\bar y$ — предельная точка последовательности y^l. Выбирая, если это необходимо, соответствующую подпоследовательность, можно утверждать, что последовательность y^l сходится к $\bar y$. Положим $t^l=s^{k_l}$ и $\mu_l=\lambda_{k_l}$. Если $F(\bar y)=m$, теорема доказана. Предположим теперь, что $F(\bar y)<m$. Тогда в силу леммы 1 существуют число $\delta>0$ и возможный вектор $\bar s$

в точке $\bar y$ такие, что $q_i(\bar y)^T\bar s<-\Theta_i\delta$ при $i\in I_C(\bar y)$, $a_i^T\bar s\leqslant 0$ при $i\in I_L(\bar y)$, $\bar s_j\geqslant 0$ при $j\in J^-(\bar y)$, $\bar s_j\leqslant 0$ при $j\in J^+(\bar y)$, вектор $\bar s$ ограничен и $g(\bar y)^T\bar s\geqslant\delta$. Отсюда вытекает возможность выбрать такое $\bar\varepsilon>0$, что для всех $x\in R(\bar y, \bar\varepsilon)$

1) $H(x)\subset H(\bar y)$,

2) $q_i(x)^T\bar s\leqslant-\Theta_i\frac{\delta}{2}$ при $i\in I_C(\bar y)$,

3) $g(x)^T\bar s\geqslant\frac{\delta}{2}$.

Пусть l_1 настолько велико, что $y^l\in R(\bar y, \bar\varepsilon)$ при $l\geqslant l_1$ и (если применяется AZ1) $H(y^l, \varepsilon)\subset H(\bar y)$ (это возможно, так как ε стремится к нулю). Тогда $g(y^l)^T t^l\geqslant\frac{\delta}{2}$ при $l\geqslant l_1$, так что, согласно лемме 2, $\sum\limits_{1}^{\infty}\mu_l$ сходится, т. е. существует такое $l_2>l_1$, что $x^{k_l+1}\in R(\bar y, \bar\varepsilon)$ для $l\geqslant l_2$. Следовательно, $g(x^{k_l+1})^T s^{k_l+1}\geqslant\frac{\delta}{2}$, откуда получим, что $x^{k_l+2}\in R(\bar y, \bar\varepsilon)$ для $l\geqslant l_3\geqslant l_2$, и т. д. Из следствия леммы 2 ясно, что всякий раз получаются точки $x^{k_l+h}\in R(\bar y, \bar\varepsilon)$, $h=1, 2, \ldots$, перемещением до новой граничной гиперповерхности из множества гиперповерхностей, содержащих $\bar y$. Но из AZ1 либо AZ2, из $g(x^{k_l+h})^T s^{k_l+h}\geqslant\frac{\delta}{2}$ и следствия леммы 3 следует, что такой гиперповерхности не найдется после конечного числа шагов. Мы пришли к противоречию, доказывающему теорему.

Без использования приемов, обеспечивающих сходимость, мы не смогли бы доказать невозможность $\sigma^k\geqslant\delta$ при всех k для ограниченной функции $F(x)$. Так как нормализация N5 подобна другим, нами доказана также сходимость для P2 при использовании AZ1 либо AZ2. В случае P2 приемы, обеспечивающие сходимость, не являются необходимыми, так как справедлива следующая теорема.

Теорема 2. *Если R и $F(x)$ удовлетворяют условиям* C1 *и* C3 *соответственно*[1]) *и если* $\varepsilon'=0$, *то*

---

[1]) См. примечание на стр. 111. — *Прим. ред.*

алгоритм Р2 *порождает последовательность точек* $x^k \in R$, *для которой* $\lim F(x^k) = m$.

Доказательство. Если последовательность x^k конечна, теорема очевидна, если она бесконечна и не ограничена, $\lim F(x^k) = \infty$ в силу условия С3. Предположим поэтому, что последовательность x^k бесконечна, но ограничена. Пусть $\bar{y}$ — предельная точка последовательности и $y^l = x^{k_l}$ — подпоследовательность x^k, сходящаяся к $\bar{y}$. Предположим, что $F(\bar{y}) < m$, тогда решением задачи (7.3.16) в точке $\bar{y}$ будет $x_0 = \delta > 0$. Следовательно, существует $\bar{\varepsilon} > 0$ такое, что для всех $y \in R(\bar{y}, \bar{\varepsilon})$ решением задачи (7.3.16) в точке y будет $x_0 \geqslant \delta/2$. Положим $\bar{\varepsilon} \leqslant \frac{\Theta_i \delta}{4}$, тогда для всех $y \in R(\bar{y}, \bar{\varepsilon})$ справедливы неравенства $b_i - f_i(y) = \frac{\Theta_i \delta}{4}$ для всех $i \in I_C(\bar{y})$. Выберем столь большое $l(\bar{\varepsilon})$, что $y^l \in R(\bar{y}, \bar{\varepsilon})$ для $l \geqslant l(\bar{\varepsilon})$. Тогда из (7.3.16) следует, что $q_i(y^l)^T (x_{\text{опт}} - y^l) \leqslant \frac{\Theta_i \delta}{4}$ для $l \geqslant l(\bar{\varepsilon})$ и $i \in I_C(\bar{y})$. Из $x_0(y^l) \geqslant \frac{\delta}{2}$ получим, что ряд $\sum_{1}^{\infty} \lambda_{k_l}$ сходится, так как в противном случае последовательность $F(x^k)$ не ограничена. Следовательно, λ стремится к нулю и $\lambda_{k_l} = \lambda'_{k_l}$ для достаточно больших l. После конечного числа шагов точка x^k может попадать только на нелинейные граничные гиперповерхности из множества гиперповерхностей, содержащих $\bar{y}$, но так как $q_i(y^l)^T (x_{\text{опт}} - y^l) \leqslant \Theta_i \frac{\delta}{4}$ при $i \in I_C(\bar{y})$, то на основании следствия леммы 3 это невозможно. Таким образом, $F(\bar{y}) = m$, что доказывает теорему.

7.7. Нелинейное программирование без предположений о выпуклости.

Во многих задачах математического программирования требуется найти максимальное значение функции $F(x)$ с непрерывными частными производными, определенной

в замкнутой связной области, задаваемой системой линейных и нелинейных неравенств. Возникает вопрос, что получится, если к этой „непрерывной" задаче программирования применить метод возможных направлений. Это важный вопрос, так как во многих практических задачах мы заранее не знаем, является ли функция $F(x)$ вогнутой и выпуклы ли функции $f_i(x)$. Пусть задача имеет вид

$$\max \{F(x) \mid f_i(x) \leqslant b_i, \ i \in I_1; \ a_i^T x \leqslant b_i, \ i \in I_2; \ 0 \leqslant x \leqslant c\}, \tag{7.7.1}$$

где $F(x)$ и $f_i(x)$ — функции с непрерывными частными производными, удовлетворяющие некоторым условиям регулярности.

С1: для любого $x \in R$ найдется такой вектор $s \in S(x)$, что $q_i(x)^T s < 0$, если $i \in I_1(x)$.

С3: $R_\alpha = \{x \in R \mid F(x) = \alpha\}$ ограничено для любого конечного α.

Первая трудность состоит в том, что в этой задаче может быть много локальных максимумов и в лучшем случае мы найдем один из них. В настоящий момент нет общего метода преодоления этой трудности. В некоторых практических задачах опасность попадания на локальный максимум можно уменьшить либо использованием априорных знаний о задаче, либо использованием большого числа начальных точек.

Вторая трудность заключается в том, что здесь точка x, удовлетворяющая условиям $g(x)^T s \leqslant 0$ для всех $s \in S(x)$, не обязательно является локальным максимумом $F(x)$ на R. Такая точка, называемая стационарной, может быть также локальным минимумом или седлом. Минимум может встретиться лишь в исходной точке x^0, если $g(x^0) = 0$, поэтому этот случай не имеет практического значения. Возможность попасть в седловую точку также маловероятна в практических задачах, однако она не может быть исключена.

Ясно, что если x — седловая точка, то существует по крайней мере один вектор $s \in S'(x)$ такой, что $g(x)^T s = 0$, $s \neq 0$, и в направлении s функция $F(x)$ возрастает.

Наконец, отметим, что выбор λ определяется либо условием

$$\lambda = \max \{\lambda \leqslant \lambda' \mid F(x + \lambda s) > F(x + \mu s) \ \text{для всех} \ \mu < \lambda\},$$

либо

$$\lambda = \max\{\lambda \leqslant \lambda' \mid g(x+\mu s)^T s \geqslant 0 \quad \text{для всех} \quad \mu \leqslant \lambda \text{ и } > 0 \quad \text{для всех} \quad \mu,\ 0 \leqslant \lambda_1 \leqslant \mu < \lambda\},$$

где λ_1 — некоторое число, меньшее λ. Первое условие ведет к выбору максимума $F(x+\lambda s)$ при $\lambda \leqslant \lambda'$, второе дает первый (локальный) максимум $F(x+\lambda s)$ при $\lambda \leqslant \lambda'$; его следует предпочесть, когда предполагается, что x близко к максимуму. Легко доказать, что любая предельная точка алгоритма P1 с AZ1 либо AZ2 и нормализациями N1 — N5 так же, как и предельная точка алгоритма P2, является стационарной. Действительно, доказательства, приведенные в п. 7.6, не изменятся, так как леммы 1—3 справедливы для произвольных $F(x)$ и $f_i(x)$.

Отсюда можно заключить, что метод возможных направлений можно применять в общем случае нелинейного математического программирования, однако неизвестно, будет ли полученное решение абсолютным максимумом или нет. Если задача удовлетворяет условию регулярности, требованию замкнутости и связности множества $R'_\alpha = \{x \mid x \in R,\ F(x) \geqslant \alpha\}$ для любого α и условию $g(x)^T s > 0$, если $g(x+\lambda s)^T s > 0$ для всех λ, $0 < \lambda < \bar{\lambda}$, где $\bar{\lambda}$ — некоторое положительное число, тогда метод возможных направлений дает абсолютный максимум $F(x)$ на R.

Глава 8

НОРМАЛИЗАЦИИ ВОЗМОЖНЫХ НАПРАВЛЕНИЙ

8.1. Введение.

В этой главе мы изучим различные нормализации возможных направлений, перечисленные в п. 7.3. Нормализации N1 уделено наибольшее внимание. Вычисления, порожденные нормализацией N3, незначительно отличаются от вычислений при нормализации N2. Нормализация N4 является, по-видимому, простейшей. Нормализация N5 приводит к своеобразному вычислительному процессу. Пункт 8.6 посвящен краткому сравнению различных возможных нормализаций.

8.2. Нормализация N1.

Пусть $x \in R$ и $Q = Q(x)$ — матрица со строками $q_i(x)$ при $i \in I_C(x)$, a_i при $i \in I_L(x)$, $-e_j$ при $j \in J^-(x)$ и e_j при $j \in J^+(x)$. Быть может, эта матрица содержит еще и строки, соответствующие дополнительным требованиям приемов, обеспечивающих сходимость. Тогда задача выбора направления может быть представлена в виде

$$\max\{\sigma \mid Qs + \sigma\Theta \leqslant 0, \quad -g^T s + \sigma \leqslant 0, \quad s^T s \leqslant 1\}, \tag{8.2.1}$$

где Θ — вектор с неотрицательными компонентами, причем $\Theta_i > 0$ для строк, соответствующих нелинейным ограничениям, и $\Theta_i = 0$ для строк, соответствующих линейным ограничениям.

Теорема 1. *Оптимальное решение s_1, σ_1 задачи (8.2.1) пропорционально оптимальному решению s_2, σ_2 задачи*

$$\max\{\sigma \mid Qs + \sigma\Theta \leqslant 0, \quad -g^T s + \sigma \leqslant 0, \quad s^T s + \sigma^2 \leqslant 1\}. \tag{8.2.2}$$

Доказательство. Ясно, что $s_1/\sqrt{1+\sigma_1^2}$, $\sigma_1/\sqrt{1+\sigma_1^2}$ — возможное решение (8.2.2), а $s_2/\sqrt{1-\sigma_2^2}$, $\sigma_2/\sqrt{1-\sigma_2^2}$ — возможное решение задачи (8.2.1).

Следовательно, когда $\sigma_1=0$, тогда $\sigma_2=0$, и наоборот, так что в этом случае $s_1=0$, $\sigma_1=0$ и $s_2=0$, $\sigma_2=0$ являются пропорциональными оптимальными решениями.

$\sigma_1 \geqslant \frac{\sigma_2}{\sqrt{1-\sigma_2^2}}$ и $\sigma_2 \geqslant \frac{\sigma_1}{\sqrt{1+\sigma_1^2}}$, следовательно, $\sigma_1^2 \geqslant \frac{\sigma_2^2}{1-\sigma_2^2}$ или $\frac{\sigma_1^2}{1+\sigma_1^2} \geqslant \sigma_2^2 \geqslant \frac{\sigma_1^2}{1+\sigma_1^2}$, т. е. $\sigma_2 = \frac{\sigma_1}{\sqrt{1+\sigma_1^2}}$. Теорема доказана.

Эта теорема показывает, что задача (8.2.1), так же как и (7.3.13)–(7.3.15) при нормализации N1, является задачей типа

$$\max\{p^T x \mid Px \leqslant 0, \quad x^T x \leqslant 1\}. \tag{8.2.3}$$

Здесь P — матрица, содержащая m строк. В дальнейшем изучается эта задача, являющаяся задачей выпуклого программирования с линейной оптимизируемой функцией и одним квадратичным ограничением. Пока для вектора x, удовлетворяющего условиям $Px \leqslant 0$, $x^T x \leqslant 1$, выполняется неравенство $p^T x > 0$, решение x' задачи (8.2.3) удовлетворяет равенству $(x')^T x' = 1$. Таким решением будет вектор единичной длины, принадлежащий конусу $\{x \mid Px \leqslant 0\}$ и образующий минимальный угол с вектором p, т. е. коллинеарный проекции p на конус. Отсюда следует

Теорема 2. *Если $p^T x' > 0$, то оптимальное решение x' единственно.*

Доказательство. Предположим, что x'' — еще одно оптимальное решение. Пусть $x = \frac{1}{2}(x' + x'')$. Тогда $p^T x = p^T x'$, $Px \leqslant 0$ и $x^T x < 1$, так что существует такое $\lambda > 1$, что λx — возможное решение. Но $\lambda p^T x > p^T x' = p^T x''$, т. е. x' и x'' не оптимальны.

Теорема 3. *Если x' — решение задачи (8.2.3) и $p^T x' > 0$, то для некоторого $\lambda > 0$ вектор $\lambda x'$ — решение задачи*

$$\min\{x^T x \mid Px \leqslant 0, \quad p^T x = 1\}. \tag{8.2.4}$$

Доказательство. Поскольку $Px' \leqslant 0$ и $p^T x' > 0$, то возможное решение задачи (8.2.4) существует. Выберем такое λ, что $\lambda p^T x' = 1$. Пусть x'' — решение (8.2.4), т. е. $(x'')^T x'' = \alpha^2 \leqslant \lambda^2 (x')^T x' = \lambda^2$. Если $\alpha < \lambda$, то $\frac{x''}{\alpha}$ — возможное решение (8.2.3) и $\frac{p^T x''}{\alpha} > \frac{p^T x''}{\lambda} = p^T x'$, так что x' не может быть оптимальным решением (8.2.3).

Из этой теоремы следует, что для решения задачи (8.2.3) можно применять любой метод решения задачи квадратичного программирования. Поскольку задача (8.2.3) — задача частного вида, для ее решения рекомендуется специальный метод, к описанию которого мы переходим.

Определим

$$y = -Px, \quad \text{т. е.} \quad y \geqslant 0. \tag{8.2.5}$$

Предположим, что x' — решение (8.2.3). Тогда из (2.6.6) следует, что существуют такие вектор u' и скаляр β, что

$$p = P^T u' + \beta x', \quad u' \geqslant 0, \quad \beta \geqslant 0, \quad (u')^T y' = 0. \tag{8.2.6}$$

Умножая (8.2.6) на матрицу $-P$ и полагая

$$v' = -\beta P x' = \beta y' \geqslant 0, \tag{8.2.7}$$

получим, что u' и v' удовлетворяют следующим соотношениям:

$$-PP^T u + v = -Pp, \tag{8.2.8}$$

$$u \geqslant 0, \quad v \geqslant 0, \quad u^T v = 0. \tag{8.2.9}$$

С другой стороны, если $\beta > 0$, то из (8.2.6) и (8.2.7) следует, что любое решение u, v задачи (8.2.8), (8.2.9) определяет векторы x и y, являющиеся решением (8.2.3). При $\beta = 0$ из (8.2.6) следует, что $p^T x \leqslant 0$ для любого вектора x, удовлетворяющего условию $Px \leqslant 0$, т. е. не существует возможного решения x, для которого $p^T x > 0$.

Предлагаемый метод состоит из следующих шагов:

1. Формулируем и решаем задачу (8.2.8), (8.2.9).

2. С помощью (8.2.6) вычисляем βx (нормализация не обязательна, так как мы заинтересованы лишь в ограниченности возможного направления. Поэтому необходимо, чтобы абсолютно наибольшая компонента βx не превосходила заданного числа).

Для решения (8.2.8) и (8.2.9) начнем с базисного решения $v=-Pp$, $u=0$ и выберем $v_i<0$. Если такого $v_i<0$ не существует, то u, v удовлетворяют (8.2.8) и (8.2.9). Если, например, $v_r<0$, заменяем в базисе v_r на u_r. Такая замена порождает преобразование матрицы по формулам, аналогичным формулам преобразования симплексного или двойственного симплексного метода. Вообще, если после ν итераций переменные v_i^ν принадлежат базису, а переменные u_i^ν ему не принадлежат (v_i^ν и u_i^ν состоят как из переменных v_i так и из u_i) и $v_i^\nu<0$ для некоторых i, выбираем одну из соответствующих строк и заменяем в базисе переменную v_i^ν на u_i^ν. Этот процесс основан на следующих теоремах.

Теорема 4. *Диагональные элементы преобразованной внебазисной матрицы всегда неположительны.*

Доказательство. На ν-й итерации базисом B является квадратная подматрица матрицы $(-PP^T,\ E)$. Пусть $PP^T=\begin{pmatrix} R_1 & R_3 \\ R_3^T & R_2 \end{pmatrix}$, где R_1 и R_2 — квадратные симметрические матрицы, R_3 — прямоугольная матрица соответствующих размеров и, кроме того, R_1 положительно определена (неотрицательная определенность следует из теорем 4 и 1 п. 2.3). Положим далее $B=\begin{pmatrix} -R_1 & 0 \\ -R_3^T & E_2 \end{pmatrix}$. Тогда

$$B^{-1}=\begin{pmatrix} -R_1^{-1} & 0 \\ -R_3^T R_1^{-1} & E_2 \end{pmatrix} \text{ и } B^{-1}(-PP^T,\ E)=$$

$$=\begin{pmatrix} -R_1^{-1} & 0 \\ -R_3^T R_1^{-1} & E_2 \end{pmatrix}\begin{pmatrix} -R_1 & -R_3 & E_1 & 0 \\ -R_3^T & -R_2 & 0 & E_2 \end{pmatrix}=$$

$$=\begin{pmatrix} E_1 & R_1^{-1}R_3 & -R_1^{-1} & 0 \\ 0 & R_3^T R_1^{-1} R_3 - R_2 & -R_3 R_1^{-1} & E_2 \end{pmatrix},$$

где E_1 и E_2 — единичные матрицы соответствующих размеров и $E=\begin{pmatrix} E_1 & 0 \\ 0 & E_2 \end{pmatrix}$.

Диагональными элементами преобразованной внебазисной матрицы являются диагональные элементы матриц $-R_1^{-1}$ и $(R_3^T R_1^{-1} R_3 - R_2)$. Эти матрицы в силу теорем 3 и 6 п. 2.3 неположительно определены, так что их диагональные элементы неположительны ($-R_1^{-1}$, конечно, отрицательно определена).

Теорема 5. *Если i-й диагональный элемент преобразованной внебазисной матрицы равен нулю, то соответствующий элемент правой части (преобразованного вектора ограничений) также равен нулю.*

Доказательство. Диагональные элементы матрицы $-R_1^{-1}$ всегда отрицательны, так как эта матрица отрицательно определена. Предположим, что $e_r^T(R_3^T R_1^{-1} R_3 - R_2)e_r=0$ для некоторого r, т. е. r-й диагональный элемент матрицы, заключенной в скобки, равен нулю. Пусть матрица P разбита построчно на две матрицы P_1 и P_2, так что $P_1P_1^T=R_1$, $P_2P_2^T=R_2$ и $P_1P_2^T=R_3$. Тогда $e_r^T(-P_2P_1^T R_1^{-1} P_1 P_2^T + P_2P_2^T)e_r=0$, или $e_r^T P_2(-P_1^T R_1^{-1} P_1 + E_2) P_2^T e_r = 0$. Матрица в скобках симметрическая и идемпотентная, т. е. в силу теоремы 5 п. 2.3 она неотрицательно определена. Тогда из теоремы 2 п. 2.3 следует, что

$$(-P_1^T R_1^{-1} P_1 + E_2) P_2^T e_r = 0,$$

т. е. $e_r^T(R_3^T R_1^{-1} P_1 p - P_2 p)=0$. Это выражение в точности совпадает с соответствующим элементом правой части, поскольку

$$-B^{-1}Pp=-\begin{pmatrix} -R_1^{-1} & 0 \\ -R_3^T R_1^{-1} & E_2 \end{pmatrix}\begin{pmatrix} P_1 p \\ P_2 p \end{pmatrix}=\begin{pmatrix} R_1^{-1} P_1 p \\ R_3^T R_1^{-1} P_1 p - P_2 p \end{pmatrix}.$$

Теорема доказана.

Из теорем 4 и 5 следует, что, когда $v_i^\nu < 0$, можно заменить v_i^ν на u_i^ν и при этом $u_i^{\nu+1} > 0$, так как соответствующий диагональный элемент отрицателен. Заметим, что при таком способе решения требование $u^T v = 0$ всегда удовлетворяется. Как только $v_i^\nu \geqslant 0$ для всех i, оптимальное решение найдено.

Последний вопрос состоит в том, как выбирать главную строку из числа строк с $v_i^\nu < 0$ так, чтобы метод был конечным. Мы укажем два возможных метода выбора, основанных на способах преодоления вырожденности, предложенных Чарнсом [36] и Данцигом, Орденом и Вулфом [17]. Эти способы обеспечивают конечность методов, но, по-видимому, не слишком эффективны. После доказательства конечности мы укажем еще два возможных способа выбора, не всегда обеспечивающих конечность, но зато практически более эффективных.

Пусть $A = -PP^T$ состоит из столбцов $a_{\cdot i}$, $i = 1, \ldots, m$, и пусть $a_{\cdot i}^\nu = \{B^\nu\}^{-1} a_{\cdot i}$ (может быть единичным столбцом). Пусть далее $b_{\cdot i}^\nu = \{B^\nu\}^{-1} e_i$ (может быть и неединичным столбцом) и $p^\nu = \{B^\nu\}^{-1}(-Pp)$, т. е. $p^0 = -Pp$. Рассмотрим два следующих способа выбора $v_i^\nu < 0$, если они существуют.

1. Пусть $I_0^\nu = \{i \mid p_i^\nu < 0\}$. Если множество I_0^ν пусто, задача решена; если оно состоит из одного элемента r, выберем в качестве главной r-ю строку. В противном случае рассмотрим

$$I_1^\nu = \left\{ i_1 \in I_0^\nu \,\middle|\, \frac{b_{i_1 1}^\nu}{-p_{i_1}^\nu} = \min_{i \in I_0^\nu} \frac{b_{i1}^\nu}{-p_i^\nu} \right\}.$$

Предположим, что $I_0^\nu, I_1^\nu, \ldots, I_{h-1}^\nu$ уже определены. Если I_{h-1}^ν состоит из одного элемента, выбираем соответствующую строку в качестве главной, в противном случае определим

$$I_h^\nu = \left\{ i_h \in I_{h-1}^\nu \,\middle|\, \frac{b_{i_h h}^\nu}{-p_{i_h}^\nu} = \min_{i \in I_{h-1}^\nu} \frac{b_{ih}^\nu}{-p_i^\nu} \right\}. \qquad (8.2.10)$$

Процесс продолжается до тех пор, пока будет определен единственный выбор. Так как строки матрицы взаимонезависимы, это произойдет при некотором $h \leqslant m$.

2. Определим множества I_h^ν следующим образом:

$$I_0^\nu = \{i \mid p_i^\nu < 0\},$$

$$I_h^\nu = \left\{ i_h \in I_{h-1}^\nu \,\middle|\, \frac{p_{i_h h}^\nu}{-p_{i_h}^\nu} = \min_{i \in I_{h-1}^\nu} \frac{p_{ih}^\nu}{-p_i^\nu} \right\}, \qquad (8.2.11)$$

где $p_{\cdot h}^\nu = a_{\cdot h}^\nu$ при $h \leqslant m$ и $p_{\cdot h}^\nu = b_{\cdot h-m}^\nu$ при $h > m$.

Теорема 6. *Если на каждой итерации переменная $v_{r_{\nu+1}}^\nu$, заменяемая в базисе переменной $u_{r_{\nu+1}}^\nu$, определяется одним из двух описанных способов, для решения задачи (8.2.8) и (8.2.9) достаточно конечного числа итераций.*

Доказательство. Пусть задача квадратичного программирования

$$\max \left\{ p^T P^T u - \frac{1}{2} u^T P P^T u \,\middle|\, u \geqslant 0 \right\} \qquad (8.2.12)$$

решается симплексным методом Вулфа [9], разработанным для решения задачи квадратичного программирования. При этом нам придется рассматривать задачу

$$\max \{\lambda \mid -PP^T u + \lambda Pp + v = 0, \; u \geqslant 0, \; v \geqslant 0, \; \lambda \leqslant 1, \; u^T v = 0\}. \qquad (8.2.13)$$

За исключением соотношения $u^T v = 0$, задача (8.2.13) является задачей линейного программирования, для которой очевидно исходное базисное решение $v = 0$, $\bar{\lambda} = 1 - \lambda = 1$, $u = 0$, $\lambda = 0$, удовлетворяющее условию $u^T v = 0$. Метод Вулфа решения (8.2.13) состоит в применении симплексного метода с дополнительным правилом исключения, запрещающим для каждого i вводить в базис u_i (v_i), если v_i (u_i) принадлежит базису. Это правило ограничивает выбор главных столбцов. Несмотря на такое ограничение, оптимальное решение $\lambda = 1$ достигается за конечное число шагов в предположении, что применяется какой-либо способ преодоления вырожденности. Соответствующие векторы u и v являются решением (8.2.8), (8.2.9). В этой специальной задаче переменная λ вводится в базис на первой итерации. Дополнительная переменная $\bar{\lambda}$ остается в базисе до последней итерации.

Как только она исключается из базиса, значение λ делается равным единице, и задача решена. Пока $\bar\lambda$ принадлежит базису, базисное значение λ равно нулю. До последней итерации все базисные переменные, исключая $\bar\lambda$, в том числе и переменная, соответствующая главной строке, равны нулю. Поэтому необходимо применять какой-либо способ преодоления вырожденности. Остановимся на втором способе п. 3.2, предложенном Данцигом, Орденом и Вулфом [17]. Пусть $\bar p^{\nu}_{\cdot s}$ — главный столбец, т. е. один из столбцов $\bar p^{\nu}_{\cdot j}$ преобразованной матрицы $(-PP^T, E)$. Для выбора направляющего элемента рассмотрим множества $I_0^\nu \supset I_1^\nu \supset \ldots \supset I_h^\nu$, где

$$I_0^\nu=\{i\mid \bar p^\nu_{is}>0\},\quad I_h^\nu=\left\{i_h\in I^\nu_{h-1}\,\middle|\,\frac{\bar b^\nu_{i_h h}}{\bar p^\nu_{i_h s}}=\min_{i\in I^\nu_{h-1}}\frac{\bar b^\nu_{ih}}{\bar p^\nu_{is}}\right\},$$

а $\bar b^\nu_{\cdot h}=\bar p^\nu_{\cdot m+h}$, $h=1, 2, \ldots$. Если I_0^ν пусто, из базиса исключается $\bar\lambda$. Если оно непусто, построим множество I_h^ν, $h\leqslant m$, состоящее из одного элемента, так что выбор направляющего элемента всегда единствен. После первой итерации все базисы будут содержать переменные λ, $\bar\lambda$ и $(m-1)$-ю переменную u_i или v_i, в то время как $(m+1)$-я переменная u_i или v_i не принадлежат базису. Отсюда и из правила исключения следует, что в точности для одного r_ν обе переменные u_{r_ν} и v_{r_ν} не принадлежат базису в конце ν-й итерации, $\nu=1, 2, \ldots$. Одна из них исключена из базиса на ν-й итерации, другая будет введена в базис на $(\nu+1)$-й итерации, так как правило исключения не дает возможности сделать иной выбор. Предположим теперь, что мы проделали ν итераций решения (8.2.8), (8.2.9), используя (8.2.10) для выбора множеств, определяющих главную строку. Будем рассматривать умноженную на -1 правую часть как дополнительный столбец внебазисной матрицы. Сопоставим ему переменную λ и проведем искусственную итерацию с $-p^\nu_{r_\nu}$ в качестве направляющего элемента. Тогда переменная u_{r_ν} или v_{r_ν}, только что введенная в базис, снова покинет его, а ее место займет λ. При этом мы получим таблицу с внебазисными u_{r_ν} и v_{r_ν} и переменной λ на r_ν-м месте базиса. Суть состоит в том, что эта таблица в точности

совпадает с таблицей метода Вулфа после ν-й итерации, нет лишь строки, соответствующей $\bar\lambda$. Если $\tilde a^\nu_{ij}$, $\tilde b^\nu_{ij}$ — элементы новой таблицы, то

$$\tilde a^\nu_{ij}=a^\nu_{ij}-a^\nu_{r_\nu j}\frac{p^\nu_i}{p^\nu_{r_\nu}},\quad i\neq r_\nu;\quad \tilde a^\nu_{r_\nu j}=-\frac{a^\nu_{r_\nu j}}{p^\nu_{r_\nu}},$$

$$\tilde b^\nu_{ij}=b^\nu_{ij}-b^\nu_{r_\nu j}\frac{p^\nu_i}{p^\nu_{r_\nu}},\quad i\neq r_\nu;\quad \tilde b^\nu_{r_\nu j}=-\frac{b^\nu_{r_\nu j}}{p^\nu_{r_\nu}},$$

и мы должны доказать, что $\bar a^\nu_{ij}=\tilde a^\nu_{ij}$ и $\bar b^\nu_{ij}=\tilde b^\nu_{ij}$. Это верно, если в обеих таблицах одни и те же переменные входят в базис, т. е. если в обоих методах выбираются одинаковые переменные $v^{\nu-1}_{r_\nu}$, исключаемые из базиса. На первой итерации метода Вулфа в базис вводится λ и r_1 определяется множествами

$$I_0^0=\{i\mid p_i^0<0\},\quad I_h^0=\left\{i_h\in I^0_{h-1}\,\middle|\,\frac{(e_h)_{i_h}}{-p^0_{i_h}}=\min_{i\in I^0_{h-1}}\frac{(e_h)_i}{-p^0_i}\right\}.$$

В нашем методе выбор в точности такой же, так что переменные v_{r_1} одинаковы в обоих методах и $\bar a^1_{ij}=\tilde a^1_{ij}$, $\bar b^1_{ij}=\tilde b^1_{ij}$. Предположим теперь, что $\bar a^\nu_{ij}=\tilde a^\nu_{ij}$, $\bar b^\nu_{ij}=\tilde b^\nu_{ij}$, т. е. в обеих таблицах для одинакового значения r_ν как u_{r_ν}, так и v_{r_ν} не принадлежат базису. В нашем методе u_{r_ν} и v_{r_ν} на ν-й итерации меняются местами. В методе Вулфа переменная u_{r_ν} (v_{r_ν}) будет введена в базис на $(\nu+1)$-й итерации, если v_{r_ν} (u_{r_ν}) исключена из него на ν-й итерации. В методе Вулфа $r_{\nu+1}$ определяется, таким образом, с помощью множеств

$$I_0^\nu=\{i\mid \bar p^\nu_{is_{\nu+1}}>0\}$$

($\bar p^\nu_{is_{\nu+1}}$ означает либо $\bar a^\nu_{ir_\nu}$, либо $\bar b^\nu_{ir_\nu}$ в зависимости от того, вводится ли в базис u_{r_ν} или v_{r_ν}),

$$I_h^\nu=\left\{i_h\in I^\nu_{h-1}\,\middle|\,\frac{\bar b^\nu_{i_h h}}{\bar p^\nu_{i_h s_{\nu+1}}}=\min_{i\in I^\nu_{h-1}}\frac{\bar b^\nu_{ih}}{\bar p^\nu_{is_{\nu+1}}}\right\}.$$

В нашем методе это суть множества

$$I_0^\nu = \{i \mid p_i^\nu < 0\};\quad I_h^\nu = \left\{ i_h \in I_{h-1}^\nu \,\middle|\, \frac{b_{i_h h}^\nu}{-p_{i_h}^\nu} = \min_{i \in I_{h-1}^\nu} \frac{b_{ih}^\nu}{-p_i^\nu} \right\}.$$

Но $\tilde b_{ij}^\nu = \tilde{\tilde b}_{ij}^\nu = b_{ij}^\nu - b_{r_\nu j}^\nu \frac{p_i^\nu}{p_{r_\nu}^\nu}$ $(i \neq r_\nu)$ и $\tilde p_{is_{\nu+1}}^\nu = \tilde p_{ir_\nu}^\nu = -\frac{p_\nu^i}{p_{r_\nu}^\nu}$ $\left(\tilde p_{ir_\nu}^\nu = \tilde a_{ir_\nu}^\nu \text{ либо } \tilde p_{ir_\nu}^\nu = \tilde b_{ir_\nu}^\nu\right)$, поэтому

$$\min_i \frac{\tilde b_{ih}^\nu}{\tilde p_{is_{\nu+1}}^\nu} = \min_i \frac{b_{ih}^\nu - b_{r_\nu h}^\nu \frac{p_i^\nu}{p_{r_\nu}^\nu}}{-\frac{p_i^\nu}{p_{r_\nu}^\nu}} = p_{r_\nu}^\nu \min_i \frac{b_{ih}^\nu}{-p_i^\nu} + b_{r_\nu h}^\nu.$$

Поскольку всегда $p_{r_\nu}^\nu > 0$ и $\tilde p_{is_{\nu+1}}^\nu = -\frac{p_i^\nu}{p_{r_\nu}^\nu}$, множества I_h^ν совпадают в обоих случаях, выбор $r_{\nu+1}$ одинаков и

$$\tilde a_{ij}^{\nu+1} = \tilde{\tilde a}_{ij}^{\nu+1}, \quad \tilde b_{ij}^{\nu+1} = \tilde{\tilde b}_{ij}^{\nu+1}$$

для всех $\nu \geqslant 0$. Следовательно, в нашем методе на каждой итерации из базиса выводится та же переменная, что и в методе Вулфа, т. е. конечность нашего метода следует из конечности метода Вулфа. Теорема 6 доказана, таким образом, для случая, когда для преодоления вырожденности применяется способ (8.2.10). При решении задачи (8.2.12) методом Вулфа способ (8.2.11) эквивалентен способу преодоления вырожденности, предложенному Чарнсом.

В ряде задач выбора направления для некоторых i в (8.2.3) должно быть $y_i = 0$ (см. гл. 9 – 11). Тогда для соответствующих значений i в задаче (8.2.8), (8.2.9) $v_i = 0$, а на переменную u_i ограничений не наложено. В этом случае можно начать решение, исключая из базиса все такие v_i, после чего соответствующие строки никогда не выбираются в качестве направляющих.

Рассмотрим теперь некоторые вычислительные аспекты нормализации N1.

1. Хотя на основании теоремы 6 можно так выбирать переменные $v_{r_{\nu+1}}^\nu$, что обеспечивается конечность метода, полученный таким образом выбор практически не наилучший. Имеются иные способы выбора.

a. $\min_i \{p_i^\nu \mid p_i^\nu < 0\}$.

b. $\min_i \left\{ \frac{p_i^\nu}{-p_{ii}^\nu} \,\middle|\, p_i^\nu < 0 \right\}$,

где p_{ii}^ν — диагональный элемент внебазисной матрицы на ν-й итерации.

Исключая, быть может, специально подобранные случаи, эти способы выбора ведут к меньшему числу притом более простых итераций.

2. Предположим, что после некоторого числа итераций переменные u_i при $i \in I_1 \subset I$ принадлежат базису, а при $i \in I_2$ не принадлежат ему. Не уменьшая общности, можно принять, что I_1 состоит из первых m_1 значений i $(m_1 < m)$. Из доказательства теоремы 4 следует, что преобразованная внебазисная матрица всегда может быть представлена в виде

$$\begin{pmatrix} -R_1^{-1} & R_1^{-1}R_3 \\ -R_3^T R_1^{-1} & R_3^T R_1^{-1} R_3 - R_2 \end{pmatrix}.$$

Следовательно,

$$p_{ij}^\nu = p_{ji}^\nu \quad \text{при} \quad i \in I_1 \text{ и } j \in I_1 \quad \text{либо} \quad i \in I_2 \quad \text{и} \quad j \in I_2,$$
$$p_{ij}^\nu = -p_{ji}^\nu \quad \text{при} \quad i \in I_1 \text{ и } j \in I_2 \quad \text{либо} \quad i \in I_2 \quad \text{и} \quad j \in I_1. \tag{8.2.14}$$

Поэтому не нужно хранить всю внебазисную таблицу, достаточно хранить лишь часть ее, расположенную ниже главной диагонали. Предполагая, что таблица хранится по столбцам, и считая известным главный столбец $p_{\cdot s}^{\nu-1}$ (т. е. и главную строку), получим следующую схему вычислений.

a. Вычислим p_i^ν: $p_i^\nu = p_i^{\nu-1} - p_s^{\nu-1} \frac{p_{is}^{\nu-1}}{p_{ss}^{\nu-1}}$, $i \neq s$; $p_s^\nu = \frac{p_s^{\nu-1}}{p_{ss}^{\nu-1}}$.

b. Если $p_i^\nu \geqslant 0$ для всех i, задача решена.

Если $p_i^\nu < 0$ для некоторого i и номер следующей главной строки выбран по правилу 1a, найдем

$$p_r^\nu = \min_i \left\{ p_i^\nu \mid p_i^\nu < 0 \right\}$$

(следовательно, $r_\nu = s_\nu = s$ и $r_{\nu+1} = s_{\nu+1} = r$). Если номер следующей главной строки выбран по правилу 1b, вычислим

$$p_{ss}^\nu = \frac{1}{p_{ss}^{\nu-1}}, \quad p_{ii}^\nu = p_{ii}^{\nu-1} - p_{si}^{\nu-1}\frac{p_{is}^{\nu-1}}{p_{ss}^{\nu-1}}, \quad i \neq s.$$

(Если i и s оба принадлежат $I_1^{\nu-1}$ или $I_2^{\nu-1}$, то $p_{si}^{\nu-1} = p_{is}^{\nu-1}$, в противном случае $p_{si}^{\nu-1} = -p_{is}^{\nu-1}$.) Определим r из условия

$$\frac{p_r^\nu}{-p_{rr}^\nu} = \min_i \left\{ \frac{p_i^\nu}{-p_{ii}^\nu} \,\middle|\, p_i^\nu < 0 \right\}.$$

c. Преобразуем часть внебазисной матрицы, расположенную ниже главной диагонали. Если $i < r$, выберем одновременно r-й элемент, если $i = r$, выберем оставшуюся часть следующего главного столбца [постоянно имея в виду (8.2.14)]. Таким образом, мы экономим не только память, но сокращаем количество умножений, вычисляя только один из двух элементов p_{ij}^ν и p_{ji}^ν. Диагональные элементы лучше всего хранить отдельно (это особенно важно, если для выбора следующей главной строки применяется правило 1b).

3. В общей задаче нелинейного программирования приходится последовательно решать ряд задач выбора направления. Соседние задачи последовательности отличаются друг от друга в следующих отношениях.

a. $i \in I(x^k)$, $i \notin I(x^{k+1})$. В этом случае $v_i' > 0$ в конечном решении k-й задачи, и можно просто исключить i-е столбец и строку.

b. К (8.2.3) добавляются новые ограничения. При этом к матрице $-PP^T$ в (8.2.8) добавляются строки и столбцы. Из расширенной исходной таблицы и оптимального базиса B предыдущей задачи немедленно получается хороший базис $\begin{pmatrix} B & 0 \\ Q & E \end{pmatrix}$ новой задачи; здесь Q — часть вновь вычисленных строк исходной таблицы, принадлежащая переменным конечного базиса предыдущей задачи. Поскольку матрица B^{-1}, обратная конечному базису предыдущей задачи, содержится в следующей задаче, можно с помощью формул

$$\begin{pmatrix} B & 0 \\ Q & E \end{pmatrix}^{-1} = \begin{pmatrix} B^{-1} & 0 \\ -QB^{-1} & E \end{pmatrix} \tag{8.2.15}$$

найти расширенную обратную матрицу, расширенную преобразованную таблицу и новые элементы правой части. С этой таблицы как исходной можно начать решение новой задачи выбора направления.

c. Некоторые строки матрицы P изменяются, так, $q_i(x^{k+1}) \neq q_i(x^k)$, $i \in I(x^k)$, $i \in I(x^{k+1})$, либо $g(x^{k+1}) \neq g(x^k)$ (см. п. 7.5). Если изменяется i-я строка, можно добавить новую строку и новый столбец к последней таблице предыдущей задачи способом, описанным в 3b. При этом следует различать три случая:

I. В конечной таблице предыдущей задачи v_i принадлежит базису. Тогда можно просто опустить i-е строку и столбец.

II. В конечной таблице в базис входит u_i и $p_{ii}' < 0$. Можно сделать одну итерацию, заменяя в базисе u_i на v_i, после чего i-е строка и столбец могут быть опущены.

III. В конечной таблице в базис входит u_i и $p_{ii}' = 0$. Сразу исключить u_i из базиса нельзя, нужно сначала гарантировать, что базисное значение u_i останется равным нулю. Как только это будет сделано, удалим u_i на следующей итерации, после чего i-е строка и столбец могут быть опущены. Ясно, однако, что трудно ожидать вычислительных преимуществ такого подхода по сравнению с полным возобновлением вычислений.

Задача (8.2.8), (8.2.9) может также решаться „униварнантным" методом Хилдрета [35]. Итерационный метод Хилдрета предназначен для максимизации вогнутой функции n переменных, на которые наложено требование неотрицательности. Этот метод является также методом возможных направлений. На k-м шаге полагаем $s^k = \pm e^l$, где $k = l \pmod n$, знак выбирается так, чтобы обеспечить условие $g(x^k)^T s^k \geqslant 0$. Если $g(x^k)^T s^k > 0$, определяем λ_k (оно может быть нулем)

и x^{k+1}. Здесь не нужно выбирать направления, поэтому алгоритм очень прост.

К сожалению, итерационный процесс медленно сходится. Вместо этого метода можно использовать следующий итерационный метод, основанный на том, что, когда векторы x', y' и u' удовлетворяют условиям (8.2.6), (8.2.7) и $\beta=1$, x' должен быть пропорциональным решению (8.2.3).

Алгоритм состоит в следующем:

1. Начнем с точки $x^0=p$ и $u^0=0$.
2. Выберем $l=1$, $\nu=1$.
3. Вычислим $y_l^{\nu-1}=-p_{l.}^T x^{\nu-1}$ ($p_{l.}$ — l-я строка матрицы P).
4. Если $y_l^{\nu-1}=0$ или $y_l^{\nu-1}>0$ и $u_l^{\nu-1}=0$, перейдем к шагу 6.

Если $y_l^{\nu-1}>0$ и $u_l^{\nu-1}>0$, положим $r_\nu=l$ и

$$\lambda_\nu=-\min\left(-\frac{p_{r_\nu.}^T x^{\nu-1}}{p_{r_\nu.}^T p_{r_\nu.}},\quad u_{r_\nu}\right).$$

Если $y_l^{\nu-1}<0$, положим $r_\nu=l$ и

$$\lambda_\nu=\frac{p_{r_\nu.}^T x^{\nu-1}}{p_{r_\nu.}^T p_{r_\nu.}}.$$

5. Вычислим

$$x^\nu=x^{\nu-1}-\lambda_\nu p_{r_\nu.},$$

$$u^\nu=u^{\nu-1}+\lambda_\nu e_{r_\nu.}.$$

6. Увеличим на единицу значения ν и l (по модулю n) и повторим шаги 3—6.

Легко проверить, что значение $x^T x$ монотонно убывает. Действительно, λ_ν в этом алгоритме совпадает с λ_ν в методе Хилдрета, откуда немедленно следует сходимость первого. Преимущество этого алгоритма заключается в том, что не требуется сначала вычислять матрицу PP^T, содержащую обычно существенно больше, чем матрица P, отличных от нуля элементов. С другой стороны, приходится вычислять скалярные произведения $p_i^T x$, а при вычислениях по методу Хилдрета они известны.

Возможность выбора подходящих возможных направлений решением задач типа (8.2.3) и эквивалентность задачи (8.2.3) задаче квадратичного программирования была независимо показана Деннисом [19, гл. 7].

8.3. Нормализация N2.

Пусть $x\in R$, тогда задача выбора направления состоит в следующем:

$$\max\{\sigma \mid q_i(x)^T s+\Theta_i\sigma\leqslant 0,\ i\in I_C(x);\ a_i^T s\leqslant 0,\ i\in I_L(x);$$
$$-g(x)^T s+\sigma\leqslant 0;\ 0\leqslant s_j\leqslant 1,\ j\in J^-(x);$$
$$-1\leqslant s_j\leqslant 0,\ j\in J^+(x);\ -1\leqslant s_j\leqslant 1,\ j\in J_1(x)\},\qquad (8.3.1)$$

где

$$J_1(x)=J-(J^-(x)+J^+(x)).$$

Если ограничить значение σ снизу нулем, а сверху $\sum|g_j|$ и заменить σ на $\dfrac{\sigma}{\sum|g_j|}$, то задача (8.3.1) превратится в задачу типа (6.2.6) с двусторонними ограничениями на переменные. Ее можно решить, как показано в п. 6.2, способом, основанным на двойственном симплексном методе, но мы предпочитаем перейти к двойственной задаче и применить алгоритм решения задач с абсолютными значениями. При этом получаем задачу типа (6.3.2)

$$\max\Big\{-\sum_{j\in J_1(x)}|v_j|-\sum_{j\in J^-(x)}\max(v_j,0)+\sum_{j\in J^+(x)}\min(v_j,0)\ \Big|\ \sum_{i\in I_C(x)}u_i q_i(x)+$$
$$+\sum_{i\in I_L(x)}u_i a_i-u_0 g(x)+v=0;$$
$$\sum_{i\in I_C(x)}u_i\Theta_i+u_0=1,\ u_0\geqslant 0,\ u_i\geqslant 0\Big\}.\qquad (8.3.2)$$

К этой задаче можно непосредственно применить алгоритм, разработанный в п. 6.3. В последней части п. 6.3 было показано, как перейти от одной задачи выбора направления к следующей. Применение модифицированного мультипликативного алгоритма обычно обеспечивает простоту и краткость вычислений.

8.4. Нормализация N3.

Здесь задача выбора направления записывается в виде

$$\max \{\sigma | q_i(x)^T s + \Theta_i\sigma \leqslant 0, \ i \in I_C(x); \ a_i^T s \leqslant 0,$$
$$i \in I_L(x); \ -g(x)^T s + \sigma \leqslant 0;$$
$$0 \leqslant s_j \leqslant 1 \ \text{при} \ j \in J^-(x) \ \text{и} \ g_j(x) > 0;$$
$$0 \leqslant s_j \quad \text{при} \ j \in J^-(x) \ \text{и} \ g_j(x) \leqslant 0;$$
$$-1 \leqslant s_j \leqslant 0 \ \text{при} \ j \in J^+(x) \ \text{и} \ g_j(x) < 0; \qquad (8.4.1)$$
$$s_j \leqslant 0 \quad \text{при} \ j \in J^+(x) \ \text{и} \ g_j(x) \geqslant 0;$$
$$s_j \leqslant 1 \quad \text{при} \ j \in J_1(x) \ \text{и} \ g_j(x) > 0;$$
$$-1 \leqslant s_j \quad \text{при} \ j \in J_1(x) \ \text{и} \ g_j(x) < 0\}.$$

Снова это задача с двусторонними ограничениями, но многие переменные ограничены лишь с одной стороны. С помощью простых преобразований ее можно свести к задаче, в которой все переменные неотрицательны и лишь некоторые ограничены сверху. Таким образом, решать эту задачу несколько легче, чем задачу (8.3.1) при нормализации N2. Если перейти к задаче, двойственной (8.4.1), получим те же ограничения, что и в задаче (8.3.2), и дополнительно

$$v_j \geqslant 0 \ \text{при} \ g_j(x) \geqslant 0 \quad \text{и} \quad j \in J^+(x) + J_1(x),$$
$$v_j \leqslant 0 \ \text{при} \ g_j(x) \leqslant 0 \quad \text{и} \quad j \in J^-(x) + J_1(x).$$

Оптимизируемая функция записывается в виде

$$-\sum_{j \in J'} \max(v_j, 0) + \sum_{j \in J''} \min(v_j, 0),$$

где

$$J' = \{j \in J^-(x) + J_1(x) | g_j(x) > 0\}$$

и

$$J'' = \{j \in J^+(x) + J_1(x) | g_j(x) < 0\}.$$

Для решения этой двойственной задачи можно разработать процедуру, почти аналогичную алгоритму решения задачи с абсолютными значениями, описанному в п. 6.3.

Замечание. При нормализации N3 может случиться, что последовательность подходящих возможных векторов не

...чена. Так как в алгоритме P1 требуется ограниченность ... п. 7.6), то можно вместо s рассматривать λs, если абсо... наибольшая компонента s превосходит заданный пре... Число $\lambda < 1$ определяется так, чтобы абсолютно наи... компонента λs в точности равнялась этому пределу.

8.5. Другие нормализации.

Нормализация N4: либо $\sigma \leqslant 1$, либо $g(x)^T s \leqslant 1$, ...м все ограничения линейны.

Задача выбора направления принимает следующий вид:

$$\max \{\sigma | q_i(x)^T s + \Theta_i\sigma \leqslant 0, \ i \in I_C(x); \ a_i^T s \leqslant 0,$$
$$i \in I_L(x); \ -g(x)^T s + \sigma \leqslant 0; \ s_j \geqslant 0, \ j \in J^-(x); \qquad (8.5.1)$$
$$s_j \leqslant 0, \ j \in J^+(x); \ \sigma \leqslant 1\}.$$

Эта нормализация весьма проста, так как к таблице добавляется лишь одно ограничение.

Переменные s_j при $j \in J_1(x) = J - (J^-(x) + J^+(x))$ свободны, т. е. могут быть отрицательными. Для решения (8.5.1) мы предлагаем следующий специальный метод (по существу являющийся симплексным).

1. Строим исходную таблицу, содержащую строки $(q_i(x)^T, \Theta_i)$, $(-g(x)^T, 1)$ и $(a_i^T, 0)$ внебазисной матрицы. Ограничение $-g(x)^T s + \sigma \leqslant 0$ будет рассматриваться как одно из ограничений $q_i(x)^T s + \Theta_i\sigma \leqslant 0$, так что множество $I_C(x)$ расширяется на один элемент. Обозначим через t_i дополнительные переменные этих ограничений. Опустим члены правой части, целиком состоящей из нулей, ограничение $\sigma \leqslant 1$ и исходную d-строку. Введем в рассмотрение множества

$$t(I) = \{t_i | i \in I(x) = I_C(x) + I_L(x)\}; \quad s(J^+) = \{s_j | j \in J^+(x)\};$$
$$s(J^-) = \{s_j | j \in J^-(x)\} \ \text{и} \ s(J_1) = \{s_j | j \in J_1(x)\}.$$

2. На первой итерации введем в базис переменную σ. Переменная, исключаемая из базиса, будет определена способом, описанным в 3b.

3. Предположим, что проведено некоторое число итераций и получена внебазисная таблица со столбцами $a^*_{.h}$,

Переменная σ принадлежит базису, а соответствующая строка таблицы состоит из элементов $a^*_{\sigma h}$. Внебазисные переменные обозначим через s^*_h, базисные — через t^*_i. Алгоритм решения состоит в следующем:

a. Выбор переменной s^*_p, которую следует ввести в базис.

(I) Определяем $H^* = \{h \mid a^*_{\sigma h} \neq 0$ и $s^*_h \in s(J_1)$, либо

$a^*_{\sigma h} < 0$ и $s^*_h \in s(J^-) + t(I)$, либо

$a^*_{\sigma h} > 0$ и $s^*_h \in s(J^+)\}$.

(II) Если $H^* = 0$, то $s = 0$ — оптимальное решение задачи (8.5.1).

(III) Если $H^* \neq 0$, определим такое $p \in H^*$, что $|a^*_{\sigma p}| \geqslant |a^*_{\sigma h}|$ для всех $h \in H^*$.

b. Выбор переменной t^*_r, которую следует исключить из базиса.

(I) Определяем $I^* = \{i \mid -a^*_{ip} \operatorname{sgn} a^*_{\sigma p} > 0$ и $t^*_i \in t(I) + s(J^-)$

либо $-a^*_{ip} \operatorname{sgn} a^*_{\sigma p} < 0$ и $t^*_i \in s(J^+)\}$.

(II) Если $I^* \neq 0$, выбираем такое $r \in I^*$, что $|a^*_{rp}| \geqslant |a^*_{ip}|$ для всех $i \in I^*$, и производим преобразование. Во время преобразования выбираем главный столбец следующей итерации.

(III) Если $I^* = 0$, решение прекращается и в качестве подходящего возможного направления выбирается

$$s^*_p = -\lambda \operatorname{sgn} a^*_{\sigma p};$$
$$s^*_j = 0, \quad j \neq p;$$
$$t^*_i = \lambda a^*_{ip} \operatorname{sgn} a^*_{\sigma p};$$
$$\sigma = \lambda a^*_{\sigma p} \operatorname{sgn} a^*_{\sigma p},$$

где λ можно определить так, чтобы $\sigma = 1$, т. е. $\lambda = \frac{1}{|a^*_{\sigma p}|}$.

Так как векторы должны быть ограниченными, разумнее выбрать $\lambda \cdot 1$ таким образом, чтобы абсолютно наибольшая компонента вектора s не превосходила заданных границ.

Указанный алгоритм основан на следующих фактах:

a. Поскольку σ — единственная переменная, от которой зависит оптимизируемая функция, всегда $d^*_h = a^*_{\sigma h}$. Эта переменная не исключается из базиса после того, как была введена туда на первой итерации (множество I^* из правила 3b никогда

не содержит номера строки, соответствующей переменной σ). Поэтому правила 3a, (I) и (II) являются обычными правилами выбора главного столбца в случае, когда некоторые переменные не ограничены, а некоторые ограничены сверху, а не снизу.

b. До тех пор пока $I^* \neq 0$, все члены правой части считаются равными нулю, т. е. r выбирается так же, как и в симплексном методе (с помощью коэффициента $-\operatorname{sgn} a^*_{\sigma p}$ выражается стремление увеличить s^*_p при $a^*_{\sigma p} < 0$ и уменьшить при $a^*_{\sigma p} > 0$). Положительные значения переменных $t^*_i \in s(J^+)$ не допускаются, переменные $t^*_i \in s(J_1)$ можно не рассматривать, так как на них не наложены ограничения. Неоднозначность преодолевается выбором абсолютного наибольшего из числа возможных направляющих элементов. Такой способ выбора направляющего элемента обычно применяется в машинных программах. Хотя он не гарантирует конечности во всех случаях, практически он наиболее удобен.

c. Если на некоторой итерации $I^* = 0$, то без ограничения $\sigma \leqslant 1$ мы получили бы неограниченное решение. Учитывая это соотношение, можно было бы на конечной итерации выбрать в качестве направляющего элемента $a^*_{\sigma p}$ и исключить переменную σ из базиса, положив ее равной своей верхней границе. Мы предпочитаем, однако, опустить последнюю итерацию и действовать, как описано на шаге 3b (III) алгоритма. Отсюда следует, что нормализация N4 ведет к алгоритму, эквивалентному процедуре, без какой-либо нормализации. Но в этом алгоритме процесс прекращается, если получено неограниченное решение, после чего подходящее возможное направление выбирается вдоль предельного луча, ведущего в бесконечность.

Следующая задача выбора направления аналогична данной. Поскольку мы не проводили последней итерации, правые части новых ограничений при добавлении их к конечной таблице предшествующей задачи также будут равны нулю.

Для решения задачи (8.5.1) можно применить модифицированный мультипликативный алгоритм, но если изменяется число переменных, то едва ли возможно использовать конечное множество η-векторов предыдущей задачи для решения следующей. Вместо этого можно применить двойственный модифицированный мультипликативный алгоритм к двойствен-

ной задаче (п. 4.6, с). Возможно также применение алгоритма с обратной матрицей для решения прямой задачи, поскольку этот алгоритм допускает добавление новых ограничений.

Нормализация N5. При этой нормализации принимаются во внимание ограничения, которым испытываемое решение x не удовлетворяет как равенствам. Эта нормализация ведет к задаче линейного программирования типа (7.3.16), которую можно решить с помощью симплексного метода. Можно ее также решить с помощью одного из методов возможных направлений, применяемых в этом случае к задаче линейного программирования. Глава 9 посвящена таким приложениям.

Нетрудно найти еще и другие нормализации, ведущие к новым процедурам решения задач выпуклого программирования. Например, вместо N3 можно потребовать $s_j \leqslant g_j(x)$ при $g_j(x) > 0$ и $g_j(x) \leqslant s_j$ при $g_j(x) < 0$.

8.6. Сравнение различных нормализаций.

При отсутствии опыта вычислений трудно сравнивать различные нормализации. Поэтому на основе теории мы лишь кратко укажем, чего можно ожидать от такого сравнения.

1. При нормализации N1 требуется, в общем, меньшее число шагов, чем при нормализациях N2 — N4. Причина этого состоит в том, что получаемое допустимое направление образует наименьший возможный угол с направлением градиента.

2. При нормализации N1 ожидаемый объем работы на один шаг существенно больше, чем при нормализациях N2—N4. Действительно, нам приходится регулярно вычислять матрицу PP^T, что требует большого числа умножений и дает в результате матрицу с большим числом ненулевых элементов. Усложняется также начало решения с помощью конечной таблицы предшествующей задачи выбора направления.

Если в матрице изменяется много строк, целесообразно применение модифицированного мультипликативного алгоритма, что дает возможность вновь начать решение с повторения и затратить меньше времени. Модифицированный мультипликативный алгоритм неприменим при N1, поэтому при этой нормализации труднее вновь начинать решение. Это особенно важно, если некоторые существенные ограничения нелинейны.

При нормализациях N2 — N4 ограничения $s_j \geqslant 0 (j \in J^-(x))$ и $s_j \leqslant 0 (j \in J^+(x))$ не меняют размеров задачи выбора направления, а при N1 эти ограничения увеличивают размеры задачи. В этом заключается еще один недостаток нормализации N1.

3. По-видимому, число шагов растет в порядке N2—N3—N4. В этом же порядке убывает объем вычислений на шаг. Что именно является более существенным, можно установить лишь опытным путем. Кроме того, это зависит от задачи.

4. Нормализация N5 приводит к задачам с большими размерами. В некоторых решавшихся примерах алгоритм P2 с N5 давал плохую сходимость в случае линейных ограничений и нелинейной оптимизируемой функции (оптимум не на вершине) и быструю сходимость в случае нелинейных (квадратичных) ограничений и линейной оптимизируемой функции (оптимум достигается на вершине). Конечно, следует ожидать, что нормализация N5 наиболее выгодна, если задача мало отличается от линейной, т. е. если большинство ограничений линейно, а нелинейные ограничения, так же как и оптимизируемая функция, в значительной степени состоят из линейных членов. Действительно, в этом случае при N5 меньше ожидаемое число шагов и требуется немного пересчетов вектора градиента и нормалей.

В гл. 11 будет показано, как можно ускорить сходимость в случае нелинейной оптимизируемой функции. Если нелинейных ограничений много и используется N5, едва ли можно начать решение задачи выбора направления с помощью конечной таблицы предшествующей задачи. Но при использовании модифицированного мультипликативного алгоритма можно, конечно, начинать с повторения.

Глава 9

ЗАДАЧА ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ И МЕТОДЫ ВОЗМОЖНЫХ НАПРАВЛЕНИЙ

9.1. Введение.

В этой главе рассматривается приложение методов возможных направлений к задаче линейного программирования. В п. 9.2 дается краткий обзор различных алгоритмов решения и доказывается конечность; вычислительные алгоритмы изучаются в п. 9.3. Пункт 9.4 посвящен обсуждению различных алгоритмов. Мы покажем, что симплексный метод и метод одновременного решения прямой и двойственной задач являются методами возможных направлений.

В п. 9.5 рассматривается приложение методов возможных направлений к задачам больших размеров.

9.2. Обзор алгоритмов.

Пусть задача линейного программирования задана в форме

$$\max \{p^T x \mid Ax \leqslant b, \quad 0 \leqslant x \leqslant c\}. \tag{9.2.1}$$

Строки матрицы A будем обозначать a_i^T. Если для некоторых i требуется $a_i^T x = b_i$, а не $\leqslant b_i$, то либо некоторые переменные заранее исключаются, либо каждое равенство рассматривается как два неравенства. Как только получим точку x^k, удовлетворяющую условию $a_i^T x^k = b_i$ для данного i, будем во всех задачах выбора направления полагать $a_i^T s = 0$ вместо $a_i^T s \leqslant 0$.

Алгоритм P1.

1. Начинаем с точки $x^0 \in R$ (см. п. 7.2, если такая точка неизвестна).

2. Пусть определены точки $x^0, x^1, \ldots, x^k$. Решим k-ю задачу выбора направления

$$\max \{p^T s \mid a_i^T s \leqslant 0,\ i \in I(x^k);\ s_j \geqslant 0,\ j \in J^-(x^k);$$
$$s_j \leqslant 0,\ j \in J^+(x^k);\ \text{N1, N2, N3 либо N4}\}. \tag{9.2.2}$$

3. Если $p^T s^k = 0$, x^k — оптимальное решение. Если $p^T s^k > 0$, определим

$$\lambda_k = \max \{\lambda \mid a_i^T (x^k + \lambda s^k) \leqslant b_i,\ 0 \leqslant x^k + \lambda s^k \leqslant c\}. \tag{9.2.3}$$

4. Если $\lambda_k = \infty$, решение задачи не ограничено (это возможно, когда некоторые из c_j равны бесконечности). Если $\lambda_k < \infty$, определим

$$x^{k+1} = x^k + \lambda_k s^k \tag{9.2.4}$$

и повторим шаги 2—4.

Теорема 1. *Для решения задачи* (9.2.1) *с помощью алгоритма* P1 *при нормализации* N1 *требуется конечное число шагов.*

Доказательство. Предположим, что s^k — решение (9.2.2) при N1. Пусть $a_i^T s^k = 0$ при $i = I_1(x^k) \subset I(x^k)$ и $a_i^T s^k < 0$ при $i \in I_2(x^k) = I(x^k) - I_1(x^k)$. Аналогично $s_j^k = 0$ при $j \in J_1^-(x^k) \subset J(x^k)$ и $s_j^k > 0$ при $j \in J_2^-(x^k) = J^-(x^k) - J_1^-(x^k)$; $s_j^k = 0$ при $j \in J_1^+(x^k)$ и $s_j^k < 0$ при $j \in J_2^+(x^k) = J^+(x^k) - J_1^+(x^k)$. Если множества $I(x^k)$, $J^-(x^k)$ и $J^+(x^k)$ в задаче (9.2.2) заменить соответственно множествами $I_1(x^k)$, $J_1^-(x^k)$, $J_1^+(x^k)$, то решение s^k останется прежним. Когда $0 < \lambda < \lambda_k$, имеют место соотношения $I(x^k + \lambda s^k) = I_1(x^k)$, $J^-(x^k + \lambda s^k) = J_1^-(x^k)$ и $J^+(x^k + \lambda s^k) = J_1^+(x^k)$. Следовательно, при $\lambda < \lambda_k$ выполняется условие $s^k \in S(x^k + \lambda s^k)$, в то же время $p^T s < p^T s^k$ для любого иного нормализованного $s \in S(x^k + \lambda s^k)$. Последнее следует из теоремы единственности п. 8.2. Если $\lambda_k < \infty$, т. е. значение x^{k+1} определяется формулой (9.2.4), то

$$I(x^{k+1}) \supset I(x^k + \lambda s^k),\ J^-(x^{k+1}) \supset J^-(x^k + \lambda s^k) \text{ и}$$
$$J^+(x^{k+1}) \supset J^+(x^k + \lambda s^k)$$

для всех λ, $0 < \lambda < \lambda_k$. Отсюда в свою очередь следует, что $S(x^{k+1}) \subset S(x^k + \lambda s^k)$ для всех λ, $0 < \lambda < \lambda_k$, в то время как $s^k \notin S(x^{k+1})$. Поэтому для $s^{k+1} \in S(x^{k+1})$ выполняется условие $p^T s^{k+1} < p^T s^k$. Следовательно, две задачи выбора направления не могут совпадать. Число всех подмножеств $I(x)$, $J^-(x)$ и $J^+(x)$ множеств I и J, а значит, и число

всевозможных задач выбора направления конечно, откуда следует конечность алгоритма.

При нормализациях N2, N3 либо N4 возможно $p^T s^{k+1} = p^T s^k$, так как теорема единственности не имеет места. Это равенство всегда выполняется при N4 в силу соотношения $p^T s^k = 1$ для всех k (дополнительное ограничение векторов s^k не требуется, так как в случае линейного программирования не нужны приемы, обеспечивающие сходимость). При использовании N2 и N3 эта трудность преодолевается добавлением ε^j к соответствующим компонентам p_j, где ε — малое положительное число. Если ε достаточно мало, то вектор p является комбинацией не менее чем n строк задачи выбора направления, т. е. решение такой задачи единственно. Следовательно, всегда $p^T s^{k+1} < p^T s^k$, откуда конечность метода следует так же, как и в случае нормализации N1. Добавление ε^j ведет к некоторому порядку предпочтений среди переменных s_j, устанавливаемому простыми правилами. Поэтому практически прибавление ε^j излишне. Если используется N4, такая ε-процедура не применима. Теорема 2 п. 9.4 устанавливает, однако, что алгоритм решения задачи линейного программирования при нормализации N4 полностью эквивалентен симплексному методу, поэтому здесь можно воспользоваться способами преодоления вырожденности, применяемыми в симплексном методе.

Отсюда следует

Теорема 2. *Алгоритм* P1 *при нормализации* N2, N3 *или* N4 *совместно со способами преодоления вырожденности дает решение за конечное число шагов.*

Вырожденность в случае P1 и N2, N3 либо N4, если не приняты меры ее преодоления, может привести к зигзагообразному движению между граничными гиперплоскостями. Вместо способов преодоления вырожденности можно применить способ, обеспечивающий сходимость AZ3 (см. п. 7.5).

Теорема 3. *Алгоритм* P1 *при* N2, N3 *или* N4 *и* AZ3 *приводит к решению за конечное число шагов.*

Доказательство. Прием AZ3 гарантирует, что после конечного числа шагов мы получим $p^T s^k = 0$ и должны будем заменить одно или несколько условий $n_h^T s = 0$ на $n_h^T s \leqslant 0$

...n_h равно a_i, e_j либо $-e_j$). Мы попадем, таким образом, в точку x^k, принадлежащую некоторому множеству граничных гиперплоскостей, причем $p^T x = p^T x^k$ для любой точки x, принадлежащей тому же множеству гиперплоскостей. Следовательно, продолжая вычисления, мы никогда не вернемся к тому же множеству гиперплоскостей. Из этого легко следует конечность алгоритма.

Алгоритм P2 ведет к исходной задаче линейного программирования. Поэтому P2 не дает новых методов в линейном случае.

9.3. Вычислительные аспекты.

1. В линейном случае изменения при переходе от одной задачи выбора направления к следующей всегда просты. Нам приходится добавлять или исключать некоторые ограничения. Это легко сделать. Если задача выбора направления решается двойственным методом и применяется модифицированный мультипликативный алгоритм, требуются только простейшие изменения исходной матрицы задачи выбора направления, а множество η-векторов, полученных при решении предыдущей задачи, не изменяется. Для ускорения последующих операций и увеличения точности можно периодически проводить повторения.

2. Если положить $t_i^k = -a_i^T s^k$, тогда (9.2.3) эквивалентно

$$\lambda_k = \min(\lambda_{1k}, \lambda_{2k}, \lambda_{3k}), \tag{9.3.1}$$

где

$$\lambda_{1k} = \begin{cases} \min\left\{ \dfrac{y_i^k}{-t_i^k} \,\middle|\, i \in I - I(x^k),\ t_i^k < 0 \right\}, \\ \infty \quad \text{при} \quad t_i^k \geqslant 0 \quad \text{для всех} \quad i \in I; \end{cases} \tag{9.3.2}$$

$$\lambda_{2k} = \begin{cases} \min\left\{ \dfrac{x_j^k}{-s_j^k} \,\middle|\, j \in J - J^-(x^k),\ s_j^k < 0 \right\}, \\ \infty \quad \text{при} \quad s_j^k \geqslant 0 \quad \text{для всех} \quad j \in J; \end{cases} \tag{9.3.3}$$

$$\lambda_{3k} = \begin{cases} \min\left\{ \dfrac{c_j - x_j^k}{s_j^k} \,\middle|\, j \in J - J^+(x^k),\ s_j^k > 0 \right\}, \\ \infty \quad \text{при} \quad s_j^k \leqslant 0 \quad \text{для всех} \quad j \in J. \end{cases} \tag{9.3.4}$$

Таким образом, нужно вычислить значения переменных t_i для всех строк, не входящих в задачу выбора направления. Отсюда и из того факта, что в задачах выбора направления приходится добавлять или исключать строки, следует, что исходную матрицу надо хранить построчно. Если в вычислительной машине матрица хранится на магнитной ленте и считывается для вычисления λ_{1k}, строка a_i^T может быть извлечена из накопителя, когда $t_i^k < 0$ и отношение $\frac{y_i^k}{-t_i^k}$ не превосходит наименьшего из полученных ранее. Для этих вычислений нам нужна величина y_i^k, поэтому ее необходимо вычислять на каждой итерации:

$$y_i^{k+1} = y_i^k + \lambda_k t_i^k, \qquad i = 1, \ldots, m. \tag{9.3.5}$$

Очевидно, $y_i^k = 0$ при $i \in I(x^k)$.

3. Методы возможных направлений, описанные в этом пункте, оперируют со строками, а не со столбцами. Так как во многих задачах $n > m$, эти строки могут быть относительно длинными, что приводит к неудобству их использования. Однако, разбивая столбцы на группы, можно решать последовательность ограниченных задач, как это показано в п. 9.5. Эти задачи имеют строки с меньшим числом элементов.

9.4. Обсуждение алгоритмов.

Нормализации N1 — N4 ведут к различным методам решения задач линейного программирования, состоящим из решения последовательности задач выбора направления и более простого определения длины шага. Две последовательные задачи выбора направления очень сходны, что можно использовать, начиная решение одной задачи с помощью оптимального решения предыдущей. Когда используются нормализации N2 — N4, число строк в задаче выбора направления всегда равно числу активных ограничений с $h \in I(x)$ и никогда не превосходит (обычно существенно меньше) числа строк матрицы A. При использовании N1 число строк задачи выбора направления, вообще говоря, не превосходит n.

Докажем прежде всего следующие теоремы.

Теорема 1. *Метод возможных направлений с нормализацией N3 при решении двойственной задачи линейного программирования эквивалентен методу одновременного решения прямой и двойственной задач.*

Доказательство. Будем решать задачу (3.4.1) методом одновременного решения прямой и двойственной задач, когда мы получим последовательность ограниченных прямых задач типа (3.4.4). Теперь используем метод возможных направлений с N3 для решения двойственной задачи (3.4.2), в которой на переменные u_i не наложены ограничения. Из (3.4.1) следует, что мы получим последовательность задач выбора направления типа

$$\max\left\{-b^T s \,\middle|\, -\sum a_{ij} s_i \leqslant 0,\ j \in J(u);\ -s_i \in 1,\ i \in I\right\},$$

где $J(u) = \{ j \mid v_j = 0 \}$ [см. (3.4.3)]. Задача, двойственная этой задаче выбора направления, в точности совпадает с (3.4.4), так что оба метода приводят к одинаковым ограниченным задачам. Далее, легко проверить, что $s_i' = d'(z_i) - 1$, где s' — оптимальное решение задачи выбора направления, а $d'(z_i)$ — оценка d переменной z_i в оптимальном решении (3.4.4). Сравнение формул (3.4.7) и (3.4.8) и формул (9.2.3), (9.3.2) и (9.2.4) показывает, что определение λ также полностью эквивалентно в обоих методах. Теорема доказана.

Теорема 2. *Метод возможных направлений с нормализацией N4 при исходном решении $x^0 = 0$, $y^0 = b \geqslant 0$ дает ту же последовательность допустимых точек, что и симплексный метод.*

Доказательство. Если мы начинаем с точки $x^0 = 0$, $y^0 = b$, то в первой задаче выбора направления все переменные s_j ограничены. Поскольку базисная переменная s_j исключается из базиса, только когда x_j достигает одного из своих граничных значений, ясно, что все внебазисные s_j ограничены. Следовательно, правила 3а, (I), (II) и (III), п. 8.5 в точности такие же, как в симплексном методе для задачи

с ограниченными переменными (см. п.6.2; ограничены переменные x_j, но не s_j). Задача выбора направления имеет вид

$$\max \{p^T s \mid a_i^T s \leqslant 0, \quad i \in I(x); \quad s_j \geqslant 0, \quad j \in J^-(x); \quad s_j \leqslant 0, \quad j \in J^+(x); \quad p^T s \leqslant 1\}.$$

В первой задаче $J^+(x) = 0$, $J^-(x) = J$ и $I(x) = \{i \mid b_i = 0\}$. Поскольку нет свободных переменных, правила 3b, (I) и (II), п. 8.5 выбора направляющего элемента в задаче выбора направления аналогичны правилам симплексного метода. Аналогия нарушается, когда на некоторой итерации симплексного метода направляющий элемент выбирается в строке с ненулевым значением компоненты b. Однако в этом случае наша задача выбора направления решена.

Выбор $\lambda > 0$ в симплексном методе определяется формулами (6.2.2) — (6.2.5). Но, поскольку $s_i^* = -a_{ip}^*$ — решение задачи выбора направления (звездочка означает элемент матрицы в момент, когда $I^* = 0$, p — номер главного столбца в этот момент), наш выбор длины шага по формулам (9.3.1) — (9.3.4) точно такой же.

В симплексном методе направляющий элемент выбирается в строке, определяющей λ. В случае неоднозначности можно выбрать абсолютно наибольший из возможных направляющих элементов. Такой выбор практически удобен, но не гарантирует решения задачи во всех случаях. В нашем методе строки, определяющие λ, добавляются к задаче выбора направления и на первой итерации решения новой задачи направляющий элемент выбирается в той же строке (в предположении, что в случае неоднозначности мы действуем, как и в предыдущей задаче). Теперь предположим, что выполнены $\nu - 1$ итерации и соблюдается полная аналогия, так что в конечном решении задачи выбора направления s_j принадлежит базису, если x_j входит в базис в симплексном методе. Переменная t_i входит в задачу выбора направления, если $y_i = 0$, и принадлежит базису, если базису принадлежит $y_i = 0$. Ясно, что выбор направляющего элемента в симплексном методе и методе возможных направлений идентичен до тех пор, пока в симплексном методе не возрастает значение линейной формы. Как только значение увеличивается, задача выбора направления решена. Выбор длины

шага λ снова тот же, что и на первой итерации. В симплексном методе одна из переменных x_j может быть теперь исключена из базиса. В этом случае новое значение λ определяется условиями $x_j = 0$ или $x_j = c_j$, так что к задаче выбора направления добавляется условие $s_j \geqslant 0$ (или $s_j \leqslant 0$). В следующей задаче выбора направления s_j исключается из базиса на первой итерации. Таким образом, полная эквивалентность сохраняется после ν шагов. Это доказывает теорему.

Эквивалентность будет также иметь место, если начать с произвольной вершины допустимой области. В этом случае в симплексном методе надо произвести предварительные операции, эквивалентные введению в базис всех переменных, на которые не наложены ограничения в нашем методе. Число строк задачи выбора направления всегда равно сумме числа базисных переменных x_j и числа базисных переменных y_i, равных нулю, в симплексном методе.

Теоремы 1 и 2 показывают, что симплексный метод и метод одновременного решения прямой и двойственной задач являются частными случаями более общего метода возможных направлений.

Последний метод использует симплексные преобразования базиса для выбора возможных направлений, а не для непосредственного решения задачи, поэтому в нем возможно движение внутри допустимой области. Отсюда вытекает возможность его расширения на случай нелинейного программирования.

Можно утверждать следующее:

1. Метод одновременного решения прямой и двойственной задач является градиентным методом с большим шагом[1]) в допустимой области двойственной задачи. Если исходное возможное решение двойственной задачи неизвестно, можно применить методы, описанные в п. 7.2.

2. При решении задач нелинейного программирования метод возможных направлений с нормализацией N3 можно рассматривать как распространение метода одновременного решения прямой и двойственной задач на случай нелинейного (выпуклого) программирования.

[1]) См. примечание 2 на стр. 10. — *Прим. ред.*

3. Метод возможных направлений с нормализацией N4 можно рассматривать как распространение симплексного метода на нелинейное (выпуклое) программирование.

4. Даже в случае линейного программирования метод возможных направлений с нормализацией N4 является более общим, чем симплексный метод, так как за исходную можно принять любую допустимую точку. Ясно, что это позволяет полнее использовать предварительные сведения о задаче.

5. Структурное различие между симплексным методом и алгоритмом P1 с N4 состоит в том, что в симплексном методе рассматриваются все ограничения, в то время как в методе возможных направлений оперируют лишь с существенными в данный момент вычислений ограничениями. Это приводит к большому объему работы между шагами в нашем методе, но сокращает размеры задачи выбора направления.

6. В методе возможных направлений вычисления наиболее экономичны, когда задача, двойственная к задаче выбора направления, решается с помощью модифицированного мультипликативного алгоритма (тогда в случае N2 или N3 мы будем применять прямой *ММА*, а в случае N4 — двойственный *ММА*. В случае N1 следует применять специальный алгоритм, описанный в п. 8.2. При нормализации N4 наряду с *ММА* может применяться алгоритм с обратной матрицей для решения прямой задачи выбора направления).

7. Без опыта вычислений трудно сравнивать различные методы решения задач линейного программирования, однако можно утверждать следующее:

a. В методах возможных направлений число шагов убывает обычно в порядке N4 — N3 — N2 — N1, но объем вычислений на шаг возрастает в этом же порядке. Нормализация N1 существенно более трудоемка, чем N2 — N4 (см. п. 8.6).

b. В методах возможных направлений можно использовать предварительные сведения об оптимальном решении, что определенно сокращает объем вычислений.

c. Программы вычислений симплексного метода проще, даже когда применяется модифицированный мультипликативный алгоритм.

d. Все методы наряду с оптимальным решением прямой задачи дают оптимальные значения двойственных перемен-

ных, поэтому изучение поведения оптимального решения не вызывает затруднений.

e. Методы возможных направлений, так же как и симплексный, легко распространяются на задачи с двусторонними ограничениями на переменные.

f. В методах возможных направлений при нормализациях N2 — N4, как и в симплексном методе, легко выполняется параметрическое программирование, учитывается влияние на оптимальное решение изменений вектора ограничений, оптимизируемой функции и элементов матрицы. Это следует из того факта, что конечная таблица последней задачи выбора направления совпадает с частью последней симплексной таблицы.

9.5. Задачи линейного программирования больших размеров.

В этом пункте мы кратко опишем, как метод возможных направлений можно применять для решения очень больших задач линейного программирования. Если размеры задачи велики из-за того, что m значительно превосходит n, преимущества методов возможных направлений очевидны. Действительно, можно ожидать, что многие строки будут несущественными и размеры задач выбора направления возрастают не очень сильно. В более вероятном случае, когда $n \gg m$, построчное решение невыгодно. Мы можем, однако, разбить множество J на подмножества J_1 и J_2 и построить матрицы A_1 и A_2, состоящие из столбцов $a_{\cdot j}$ при $j \in J_1$ и $j \in J_2$ соответственно. Затем решаем ограниченную задачу

$$\max \{p_1^T x_1 \mid A_1 x_1 \leqslant b, \quad x_1 \geqslant 0\}. \qquad (9.5.1)$$

Другими словами, мы предполагаем, что в оптимальном решении (9.2.1) $x_j = 0$ при $j \in J_2$. Можно также начать решение, положив $x_j = x_j^0$ при $j \in J_2$ и b заменив $b - A_2 x_j^0$. Обозначим решение ограниченной задачи через x'_{1j} и $x'_{2j} = x^0_{2j}$. Затем вычислим d-оценки переменных x_{2j}. Это легко сделать с помощью результатов решения последней задачи выбора направления. Если u'_i — двойственная переменная, соответствующая i-му ограничению, то $d'(x_{2j})$

$=\sum u'_i a_{ij} - p_{2j}$. Сделаем теперь новое разбиение J_3, J_4, например, следующим путем. Если $j \in J_2$, $x'_{2j}=0$ и $d'(x_{2j}) \leqslant 0$, или $x'_{2j}=c_j$ и $d'(x_{2j}) \geqslant 0$, или $0 < x'_{2j} < c_j$, отнесем j к J_3. Если $j \in J_2$, $x'_{2j}=0$ и $d'(x_{2j}) > 0$ или $x'_{2j}=c_j$ и $d'(x_{2j}) < 0$, отнесем j к J_4. Если $j \in J_1$ и $x'_{1j}=0$ или $x'_{1j}=c_j$ и $d'(x_{1j}) \neq 0$, отнесем j к J_4, остальные $j \in J_1$ отнесем к J_3. Таким образом, мы получим новую ограниченную задачу

$$\max\left\{ p_3^T x_3 \,\middle|\, A_3 x_3 \leqslant b - A_4 x'_4,\ x_3 \geqslant 0 \right\}. \quad (9.5.2)$$

Поскольку x'_1, x'_2 — возможное решение (9.5.2), оптимальное решение x'' задачи (9.5.2) удовлетворяет условию $p^T x'' \geqslant p^T x'$, а в предположении невырожденности — условию $p^T x'' > p^T x'$. Каждая новая ограниченная задача эквивалентна исходной при дополнительных условиях, что $x_j=0$ либо $x_j = c_j$ для некоторых значений j.

Если вырожденность преодолена, то значения линейной формы монотонно возрастают, т. е. одно и то же множество переменных, имеющих одинаковые граничные значения, не встречается дважды. Отсюда следует конечность метода. Если мы применяем модифицированный мультипликативный алгоритм и решаем задачи, двойственные к задачам выбора направления, тогда решение каждой ограниченной задачи типа (9.5.2) будем начинать с предварительных операций. Эти операции приводят к базису, содержащему те же переменные, что были в базисе последней задачи выбора направления. Применим описанную процедуру к задаче с треугольно-блочной матрицей

$$\max\left\{ p_0^T x_0 + \sum_{h=1}^{r} p_h^T x_h \,\middle|\, A_{h0} x_0 + A_{hh} x_h \leqslant b_h,\right.$$
$$\left. x_0 \geqslant 0,\ x_h \geqslant 0,\ h=1, \ldots, r \right\}, \quad (9.5.3)$$

где для каждого $h \geqslant 0$ p_h и x_h — векторы с одинаковым числом n_h компонент; b_h — m_h-мерный вектор; A_{h0} и A_{hh} — матрицы $m_h \times n_0$ и $m_h \times n_h$ соответственно. Мы предложим

теперь алгоритм, аналогичный, за исключением первого шага, ранее описанному в данном пункте.

1. Первый шаг.

a. Фиксируем значение связующих переменных x_{0j}, положив $x_{0j} = x_{0j}^0$ (чем лучше оценка x_{0j}^0, тем меньше шагов придется сделать).

b. Решим r различных малых задач линейного программирования

$$\max\left\{ p_h^T x_h \,\middle|\, A_{hh} x_h \leqslant b_h - A_{h0} x_0^0 \right\}$$

и получим конечное решение (y_h^*, x_h^*), конечную матрицу A_{hh}^*, конечную d-строку — p_h^* и обратную матрицу $\{B_{hh}^*\}^{-1}$.

c. Преобразуем A_{h0}: $A_{h0}^* = \{B_{hh}^*\}^{-1} A_{h0}$. Вычислим p_0^*.

Мы получили новую задачу линейного программирования, по-прежнему треугольно-блочную:

$$\max\left\{ (p_0^*)^T x_0 + \sum_{h=1}^{r} (p_h^*)^T x_h^* \,\middle|\, A_{h0}^* x_0 + A_{hh}^* x_h^* \leqslant b_h^*,\right.$$
$$\left. x_0 \geqslant 0,\ x_h^* \geqslant 0,\ h=1, \ldots, r \right\}. \quad (9.5.4)$$

2. Применяем к (9.5.4) метод возможных направлений. В качестве исходных выберем точку $x_0 = x_0^0$, $x_h^* = 0$ и разбиение $j \in J_1$, если j относится к переменной x_0, в противном случае $j \in J_2$. Таким образом, мы найдем лучшее x_0 в предположении, что $x_h^* = 0$.

3. Вычислим двойственные переменные и сделаем новое разбиение J_3, J_4, как было описано выше, и т. д.

Первый шаг делается так, что его конечная таблица треугольно-блочная и вместе с тем, если оценки x_0^0 хороши, в конце шага многие переменные x_{hj} принадлежат базису. Поэтому задачи выбора направления на втором и третьем шагах содержат относительно мало ограничений.

Подобные разбиения могут применяться и в других специальных задачах.

Глава 10

КВАДРАТИЧНОЕ ПРОГРАММИРОВАНИЕ

10.1. Введение.

В этой главе предлагаются некоторые конечные методы квадратичного программирования, основанные на понятиях возможных направлений и сопряженных направлений.

В п. 10.2 описаны различные алгоритмы и приведены доказательства их конечности. Пункт 10.3 посвящен обсуждению методов квадратичного программирования. В нем устанавливается также, что метод квадратичного программирования Била является методом возможных направлений.

10.2. Описание алгоритмов.

Пусть дана задача квадратичного программирования

$$\max\left\{ p^T x - \frac{1}{2} x^T C x \,\middle|\, Ax \leqslant b; \quad 0 \leqslant x \leqslant c \right\}, \tag{10.2.1}$$

где p, c и x — n-мерные векторы; C — симметрическая неотрицательно определенная матрица порядка n; A — матрица размеров m на n; b — m-мерный вектор.

Если некоторые линейные ограничения заданы в виде равенств, можно либо исключить некоторые переменные, либо при $x \in R$ требовать $a_i^T s = 0$ вместо $a_i^T s \leqslant 0$. В первом случае уменьшается число переменных, но может утратиться простая структура матриц A и C. Таким образом, от задачи зависит, какой путь выгоднее избрать.

Пусть $x \in R$. Всегда справедливо равенство

$$g(x) = p - Cx. \tag{10.2.2}$$

Пусть $s \in S(x)$ и $g(x)^T s > 0$. Если

$$\lambda' = \max\{\lambda \mid x + \lambda s \in R\},$$
$$\lambda'' = \max\{\lambda \mid g(x + \lambda s)^T s \geqslant 0\}, \tag{10.2.3}$$

т. е.

$$\lambda'' = \frac{g(x)^T s}{s^T C s} \quad \text{при} \quad s^T C s > 0, \tag{10.2.4}$$
$$\lambda'' = \infty \quad \text{при} \quad s^T C s = 0,$$

тогда

$$\lambda = \min(\lambda', \lambda''). \tag{10.2.5}$$

Далее,

$$x^{k+1} = x^k + \lambda_k s^k, \tag{10.2.6}$$
$$g(x^{k+1}) = g(x^k) - \lambda_k C s^k, \tag{10.2.7}$$

$$\Delta F(x^k) = \lambda_k g(x^k)^T s^k - \frac{1}{2}\lambda_k^2 (s^k)^T C s^k = \frac{1}{2}\lambda_k \{g(x^k)^T s^k + g(x^{k+1})^T s^k\}, \tag{10.2.8}$$

так что при $\lambda_k = \lambda_k''$

$$\Delta F(x^k) = \frac{1}{2}\lambda_k g(x^k)^T s^k. \tag{10.2.9}$$

Введем теперь в задачах выбора направления следующие дополнительные ограничения.

1. AZ3 (см. п. 7.5).

2. a. Если $\lambda_k = \lambda_k'' < \lambda_k'$, в следующей задаче выбора направления добавим требование $s^T C s^k = 0$ и сохраним все подобные ограничения текущей задачи.

b. Если $\lambda_k = \lambda_k'$, в следующей задаче выбора направления все дополнительные требования типа $s^T C s^k = 0$ опускаются.

Соотношение $s^T C s^k = 0$ является линейным (s^k — известный вектор), поэтому оно не вызывает трудностей в решении задачи выбора направления. Это соотношение эквивалентно уравнению $\{g(x^{k+1}) - g(x^k)\}^T s = 0$ и выражает условие сопряженности векторов s и s^k относительно матрицы C.

Предположим, что $\lambda = \lambda_h'' < \lambda_h'$ для $h \geqslant r$. Тогда в каждой задаче выбора направления мы добавляем ограничение вида $s^T C s^h = 0$. На основании теоремы 7 п. 2.3 векторы s^k при $h \geqslant r$ линейно независимы, т. е. после конечного числа шагов в некоторой задаче выбора направления мы получим $g^T s_{\text{опт}} = 0$. Пусть это произойдет на k-м шаге. Тогда $x^k = x' + \sum_{h=r}^{k-1} \lambda_h s^h$.

Теорема 1. *Предположим, что $\lambda_h = \lambda_h'' < \lambda_h'$ для $h \geqslant r$, так что $(s^h)^T C s^l = 0$ для $h \geqslant r, l \geqslant r, h \neq l, s^h \in S(x^h)$. Пусть $x^{h+1} = x^h + \lambda_h s^h$, и предположим, что x^k, $k > r$, удовлетворяет условию $g(x^k)^T s \leqslant 0$ для всех $s \in S(x^k)$, для которых $s^T C s^h = 0$ при $h = r, r+1, \ldots, k-1$. Тогда $g(x^k)^T s \leqslant 0$ для всех $s \in S(x^k)$.*

Доказательство. Предположим, что $g(x^k)^T s > 0$ для некоторого $s \in S(x^k)$.

Определим вектор t условиями

$$s = \sum_{h=r}^{k-1} \mu_h s^h + t, \quad \text{где} \quad \mu_h = \frac{(s^h)^T C s}{(s^h)^T C s^h}.$$

Тогда $(s^l)^T C t = 0$ при $l = r, r+1, \ldots, k-1$ и $g(x^k)^T s = g(x^k)^T t > 0$ (см. доказательство теоремы 8 п. 2.3), т. е. вектор t сопряжен с векторами s^l при $l = r, r+1, \ldots, k-1$ и $g(x^k)^T t > 0$. Более того, если $i \in I(x^k)$, то $i \in I(x^h)$ для всех h, $r \leqslant h \leqslant k-1$ (в противном случае $\lambda_h = \lambda_h'$ для некоторого h, $r \leqslant h \leqslant k-1$). Отсюда следует, что для всех $i \in I(x^k)$ справедливо $a_i^T s^h = 0$, $r \leqslant h \leqslant k-1$ [если $a_i^T s^h < 0$, мы бы покинули i-ю гиперплоскость и не вернулись на нее, т. е. включение $i \in I(x^k)$ не имело бы места]. То же самое верно и для $J(x^k)$: если $j \in J^-(x^k) + J^+(x^k)$, то $s_j^h = 0$ для $r \leqslant h \leqslant k-1$. Следовательно, $t \in S(x^k)$, т. е. вектор t определяет подходящее возможное направление, сопряженное со всеми векторами s^h, $r \leqslant h \leqslant k-1$, что противоречит предположению о том, что такого вектора не существует. Теорема доказана.

Прежде чем доказывать конечность, опишем кратко один шаг алгоритма.

1. Предположим, что точка $x^k \in R$ определена и известны результаты решения последней задачи выбора направления. Предположим далее, что определено множество Z индексов l и j и индекс $r \leqslant k-1$ (Z может быть пустым).

2. Если $\lambda_{k-1} = \lambda_{k-1}'' < \lambda_{k-1}'$, к задаче выбора направления добавим ограничение $(s^{k-1})^T C s = 0$, после чего она принимает вид

$$\max \{g(x^r)^T s \mid a_i^T s = 0,\ i \in Z;\ a_i^T s \leqslant 0,\ i \in I(x^k) - Z;\ s_j \geqslant 0,\ j \in J^-(x^k) - Z;\ s_j = 0,\ j \in Z;\ s_j \leqslant 0,\ j \in J^+(x^k) - Z;\ (s^l)^T C s = 0,\ l = r, \ldots, k-1;\ \text{N1} - \text{N4 или N5}\}. \quad (10.2.10)$$

Если $\lambda_{k-1} = \lambda_{k-1}'$ и $a_i^T x^{k-1} < b_i$, $a_i^T x^k = b_i$ и $a_i^T x^\nu = b_i$ для некоторого $\nu < k-1$, добавим i к Z; добавим j к Z, если $x_j^{k-1} > 0$, $x_j^k = 0$ и $x_j^\nu = 0$ для некоторого $\nu < k-1$ или если $x_j^{k-1} < c_j$, $x_j^k = c_j$ и $x_j^\nu = c_j$ для некоторого $\nu < k-1$; положим $r = k$ и вычислим $g(x^k)$. Задача выбора направления принимает вид

$$\max \{g(x^k)^T s \mid a_i^T s = 0,\ i \in Z;\ a_i^T s \leqslant 0,\ i \in I(x^k) - Z;\ s_j = 0,\ j \in Z;\ s_j \geqslant 0,\ j \in J^-(x^k) - Z;\ s_j \leqslant 0,\ j \in J^+(x^k) - Z;\ \text{N1—N4 или N5}\}. \quad (10.2.11)$$

3. Решим задачу выбора направления и получим s^k.

4. Если $g(x^k)^T s^k > 0$, вычислим Cs^k, λ_k' и λ_k'' с помощью формул (10.2.3) и (10.2.4). Определим $\lambda_k = \min(\lambda_k', \lambda_k'')$. Если $\lambda_k = \infty$, решение задачи не ограничено. Если $g(x^k)^T s^k = 0$ и $Z = 0$ (пустое множество), задача решена. Если $g(x^k)^T s^k = 0$ и $Z \neq 0$, заменим Z на 0 и снова решим задачу выбора направления.

5. Вычислим x^{k+1} с помощью (10.2.6) и $F(x^{k+1})$ с помощью (10.2.8).

Теорема 2. *При соблюдении принципа сопряженности решение задачи квадратичного программирования методом возможных направлений с нормализациями* N1 — N4 *или* N5 *и* AZ3 *достигается за конечное число шагов.*

Доказательство. Если $\lambda_k'' = \infty$ для некоторого k, из (10.2.4) следует, что $(s^k)^T C s^k = 0$, т. е. $Cs^k = 0$ (теорема 2 п. 2.3). Тогда $g(x^k + \lambda s^k) = g(x^k)$ для всех λ и $F(x^k + \lambda s^k) = F(x^k) + \lambda g(x^k)^T s^k$, т. е. $F(x^k + \lambda s^k)$ как функция λ не ограничена. Таким образом, если $\lambda_k = \infty$ для

некоторого k, то и $\lambda_k'' = \infty$ и функция $F(x)$ не ограничена на R, т. е. теорема справедлива. (Заметим, что условие C3 здесь не требуется.) Предположим теперь, что $\lambda_k < \infty$ для всех k. Из AZ3 и принципа сопряженности следует, что после конечного числа шагов в некоторой задаче выбора направления выполняется условие $g(x^k)^T s_{опт} = 0$. Тогда в силу теоремы 1 получен максимум $F(x)$ при дополнительном условии, что линейные ограничения из некоторого множества удовлетворяются как равенства. Если можно продолжить решение, меняя в некоторых ограничениях $a_i^T s = 0$ или $s_j = 0$ знаки равенства на знаки $\leqslant$, то значение $F(x)$ увеличится, т. е. мы никогда не возвратимся к тому же множеству граничных гиперплоскостей. Вторая остановка снова произойдет после конечного числа шагов, и т. д. Из конечности множества всех подмножеств граничных гиперплоскостей следует конечность метода.

Поскольку нормализация N5 в некоторых отношениях отличается от остальных, кратко рассмотрим алгоритм P2, основанный на N5 (алгоритм P1 строится на нормализациях N1 — N4).

1. Предположим, что точка $x^k \in R$ определена и известны результаты решения последней частной задачи линейного программирования. Предположим также, что известны множество Z индексов i и j и индекс $r \leqslant k-1$.

2. Если $\lambda_{k-1} = \lambda_{k-1}'' < \lambda_{k-1}'$, добавим к таблице ограничение $(s^{k-1})^T C (x - x^k) = 0$. Тогда новая частная задача линейного программирования может быть представлена в виде

$$\max \{ g(x^r)^T x \mid a_i^T x = b_i, \quad i \in Z;\ a_i^T x \leqslant b_i,\ i \notin Z;$$
$$0 \leqslant x_j \leqslant c_j,\ j \notin Z;\ \ x_j = x_j^k,\ j \in Z;$$
$$(s^l)^T C x = (s^l)^T C x^k,\ l = r, \ldots, k-1 \}. \quad (10.2.12)$$

Если $\lambda_{k-1} = \lambda_{k-1}'$ и $a_i^T x^{k-1} < b_i$, $a_i^T x^k = b_i$ и $a_i^T x^\nu = b_i$ для некоторого $\nu < k-1$, добавим i к Z; добавим j к Z, если $x_j^{k-1} > 0$, $x_j^k = 0$ и $x_j^\nu = 0$ для некоторого $\nu < k-1$ или если $x_j^{k-1} < c_j$, $x_j^k = c_j$ и $x_j^\nu = c_j$ для некоторого $\nu < k-1$; положим $r = k$ и вычислим $g(x^k)$. Частная задача линейного программирования принимает вид

$$\max \{ g(x^k)^T x \mid a_i^T x = b_i,\ i \in Z;\ a_i^T x \leqslant b_i,\ i \notin Z;$$
$$0 \leqslant x_j \leqslant c_j,\ j \notin Z;\ \ x_j = x_j^k,\ j \in Z \}. \quad (10.2.13)$$

3. Решим частную задачу линейного программирования и получим максимум $(x^k)'$, построим $s^k = (x^k)' - x^k$ [$(x^k)'$ может быть бесконечным, в этом случае s^k — экстремальный луч, получаемый из конечной таблицы].

4. Если $g(x^k)^T s^k > 0$, вычислим Cs^k, λ_k' и λ_k'' [$\lambda_k' = 1$, если $(x^k)'$ конечно, λ_k'' определяется (10.2.4)]. Определим $\lambda_k = \min(\lambda_k', \lambda_k'')$. Если $\lambda_k = \infty$, решение не ограничено. Если $g(x^k)^T s^k = 0$ и $Z = 0$ (пустое множество), задача решена; если $g(x^k)^T s = 0$ и $Z \neq 0$, заменим Z пустым множеством и решим задачу снова.

5. Вычислим x^{k+1} и $F(x^{k+1})$ по формулам (10.2.6) и (10.2.8).

В обоих алгоритмах P1 и P2 мы начинаем с точки $x^0 \in R$ и $Z = 0$ (см. п. 7.2, если точка x^0 неизвестна). Вводя произвольное $\lambda_{-1} = \lambda_{-1}'$, мы можем применить описанную схему для определения s^0. Таким образом, первая задача выбора направления не содержит ограничений, связанных с принципом сопряженности. Если $\lambda_{k-1} = \lambda_{k-1}'' < \lambda_{k-1}'$, вычисление градиента в следующей задаче излишне, поскольку для произвольного s, удовлетворяющего условию $(s^k)^T C s = 0$, справедливо $g(x^k)^T s = g(x^{k-1})^T s$. Поэтому в (10.2.10) и (10.2.12) вместо $g(x^k)$ в качестве направляющего вектора оптимизируемой функции можно использовать $g(x^r)$. Как только $\lambda_{k-1} = \lambda_{k-1}'$, направляющим вектором оптимизируемой функции будет вектор $g(x^k)$. Если $\lambda_{k-2} = \lambda_{k-2}'' < \lambda_{k-2}'$, вектор $g(x^{k-1})$ нам неизвестен, и для вычисления $g(x^k)$ нельзя использовать (10.2.7).

Однако имеются другие возможности. Предположим, что известны результаты решения $(k-1)$-й задачи выбора направления типа (10.2.10) или (10.2.12), в частности матрица, обратная двойственному базису [она легко получается из η-векторов, если для решения задачи, двойственной к (10.2.10) или (10.2.12), применяется прямой или двойственный модифицированный мультипликативный алгоритм]. Обозначим

двойственный базис через B. Если в этой задаче выбора направления $s^TCs^h = 0$ (s^h — известный вектор, s — искомый вектор), двойственная переменная, связанная с таким ограничением, свободна, т. е. она входит в двойственный базис, скажем, на j_h-м месте. Так как

$$g(x^{k-1}) = g(x^r) - \sum_{h=r}^{k-2} \lambda_h C s^h,$$

то

$$B^{-1} g(x^{k-1}) = B^{-1} g(x^r) - \sum_{h=r}^{k-2} \lambda_h e_{j_h},$$

и, вычитая λ_h из j_h-й компоненты $B^{-1}g(x^r)$, $h = r, \ldots, k-2$, получим $B^{-1}g(x^{k-1})$, т. е. преобразованное значение $g(x^{k-1})$. Более того, $B^{-1}g(x^k) = B^{-1}g(x^{k-1}) - \lambda'_{k-1}B^{-1}Cs^{k-1}$. Так как вектор Cs^{k-1} вычислен, необходимо лишь умножить его на η-векторы.

10.3. Обсуждение методов квадратичного программирования.

Нормализации N1 — N5 ведут к различным методам решения задач квадратичного программирования. Помимо этих пяти, хорошо известны, например, такие методы, как методы Била [4, 5], Франка и Вулфа [32], Марковица [24], Вулфа [9].

Теорема 1. *Метод квадратичного программирования Била является методом возможных направлений, удовлетворяющих принципу сопряженности. Он эквивалентен нашему методу с нормализацией* N4, *но без* AZ3.

Доказательство мы опускаем, так как хотя оно и не сложно, но требует детального изложения метода Била. Следует заметить, что метод Била полностью отличается от описанных методов по построению и порядку вычислений. В этом методе последовательно преобразуется матрица C, в то время как в нашем методе мы все время оперируем с исходной матрицей.

При нормализациях N1 — N4 задачи выбора направления невелики, но между решением задач такого рода приходится для определения длины шага дополнительно вычислять Cs.

Объем работы для решения задачи выбора направления бывает, вообще говоря, в порядке N1 — N4, но число шагов обычно растет в этом же порядке. В случае нормализации N1 уменьшение числа шагов не компенсирует, по-видимому, возрастания объема работы на один шаг, исключая, быть может, специальные случаи. Алгоритм P2 (нормализация N5) ведет обычно к дальнейшему уменьшению числа шагов, но это достигается ценой увеличения размеров задач выбора направления. Если квадратичная функция близка к линейной, т. е. содержит мало членов второго порядка, этот алгоритм, вероятно, наиболее эффективен. Если добавляется много ограничений вида $(s^k)^T Cs = 0$, то число ограничений в частных задачах линейного программирования последовательно возрастает, но оно никогда не превосходит числа $(m+n)$ и обычно существенно меньше его. Число внебазисных переменных всегда равно n.

При использовании P1 с нормализациями N2 — N4 число ограничений обычно меньше n. Оно равно сумме числа активных гиперплоскостей и числа ограничений, связанных с принципом сопряженности. Если активны многие ограничения $x_j = 0$ или $x_j = c_j$, это число существенно меньше n (исключая случай сильной вырожденности). Во всех наших методах на каждой итерации необходимо вычислять произведение Cs-матрицы на вектор. В иных методах вычислять Cs не приходится, но там больше размеры частных задач линейного программирования. Например, в методе Франка и Вулфа, так же как и в методе Вулфа, имеется $(m+n)$ ограничений и $(m+2n)$ внебазисных переменных. В методе Била число ограничений равно $(m+n)$, но число внебазисных переменных равно n.

Во всех этих методах, производя последовательное преобразование матрицы C, можно исключить операции умножения матрицы на вектор. Следует поэтому ожидать, что такие методы обладают преимуществом, если матрица C плотна (содержит много ненулевых элементов). С другой стороны, наши методы возможных направлений предпочтительнее, если оптимизируемая функция содержит лишь немного квадратичных членов, что имеет место в большинстве практических задач.

Программа вычислений в методе Вулфа самая простая, она мало отличается от программ симплексного метода линейного программирования. Метод Била и метод Франка

и Вулфа не обладают такой структурной простотой, но это компенсируется в методе Била меньшими размерами вспомогательных задач линейного программирования. Методы возможных направлений требуют еще более сложных программ, но, с другой стороны, еще меньших частных вспомогательных задач. Кроме того, возможность начинать решение с произвольной допустимой точки дает существенные вычислительные преимущества.

Глава 11

ВЫПУКЛОЕ ПРОГРАММИРОВАНИЕ

11.1. Введение.

В этой главе рассматривается общая задача выпуклого программирования. В п. 11.2 показывается, как можно улучшить сходимость алгоритмов, описанных в п. 7.5. Пункт 11.3 посвящается вычислительным аспектам алгоритмов. В п. 11.4 эти алгоритмы обсуждаются и кратко рассматриваются еще два метода максимизации вогнутой нелинейной функции при линейных ограничениях: метод Франка и Вулфа и метод проекций градиента Розена. Показывается, что они эквивалентны методам возможных направлений.

11.2. Алгоритмы выпуклого программирования.

В п. 7.5 приведены несколько алгоритмов выпуклого программирования. Эти алгоритмы различаются способами нормализации возможных направлений и приемами, обеспечивающими сходимость. В п. 7.6 было показано, что эти алгоритмы дают последовательность возможных решений x^k с монотонно возрастающими значениями оптимизируемой функции $F(x)$, сходящимися к максимуму $F(x)$ на выпуклом множестве R. Однако следует ожидать, что при сильной нелинейности оптимизируемой функции итеративный процесс сходится медленно. В случае квадратичной оптимизируемой функции, по-видимому, возможно получить конечные методы, применяя принцип сопряженных направлений и другой прием, обеспечивающий сходимость (AZ3). Применение тех же приемов в случае произвольной вогнутой оптимизируемой функции, по-видимому, улучшает сходимость.

В квадратичном случае мы добавляли ограничения $(s^k)^T Cs = 0$ к задаче выбора направления при условии, что $\lambda_k = \lambda_k'' < \lambda_k'$, и опускали их, когда $\lambda_l = \lambda_l'$ для некоторого $l > k$. Соотношение $(s^k)^T Cs = 0$ эквивалентно равенству

$\{g(x^{k+1}) - g(x^k)\}^T s = 0$, а такое ограничение можно добавлять в случае нелинейной оптимизируемой функции более общего вида. Эти ограничения исключат мелкие шаги в отдалении от оптимальной точки, порождаемые быстрым изменением градиента.

При нормализациях N1 — N3 или N4 для эффективного предотвращения подобных мелких шагов можно также применять приемы, обеспечивающие сходимость. Особенно эффективным в этом отношении представляется AZ3.

Однако этот прием трудно использовать в случае нелинейных ограничений, так как он не обеспечивает сходимости во всех возможных случаях. Поэтому мы предлагаем использовать комбинацию AZ2 и AZ3.

Предлагаются следующие два алгоритма максимизации вогнутой функции на выпуклой области.

Алгоритм P3.

1. Найдем некоторое $x^0 \in R$ (если такая точка неизвестна, см. п. 7.2), возьмем произвольное $\varepsilon_0 > 0$ и выберем столь малое $\varepsilon' > 0$, что имеет смысл прекратить решение, когда возрастание оптимизируемой функции оказывается меньше, чем ε'. Положим далее $Z = 0$ (пустое множество) и $\varepsilon'' > 0$.

2. Предположим, что определены точки $x^0, x^1, \ldots, x^k$ $(x^h \in R,\ F(x^h) > F(x^{h-1}))$, число ε_k и множество $Z \subset I_L(x^k) + J^-(x^k) + J^+(x^k)$ индексов i и j. Произведем теперь следующие операции для нахождения x^{k+1}.

a. Вычислим $g^k = g(x^k)$.

b. Если $\lambda_{k-1} = \lambda''_{k-1} < \lambda'_{k-1}$ $(\lambda_{-1} = \lambda'_{-1}$, по определению), добавим к задаче выбора направления ограничение $(g^k - g^{k-1})^T s = 0$, так что последняя примет вид

$$\max \{\sigma \mid (s, \sigma) \in S'(x^k, \varepsilon_k); \quad n_h^T s = 0, \quad h \in Z;$$
$$(g^{l+1} - g^l)^T s = 0, \quad l = r, \ldots, k-1;$$
$$-(g^k)^T s + \sigma \leqslant 0; \quad \text{N1} - \text{N3 или N4}\}, \qquad (11.2.1)$$

где множество $S'(x^k, \varepsilon_k)$ определено формулой (7.5.7); n_h равно a_i, e_j или $-e_j$, r таково, что $\lambda_{r-1} = \lambda'_{r-1}$, но $\lambda_h = \lambda''_h < \lambda'_h$ при $r \leqslant h \leqslant k-1$. Если $\lambda_{k-1} = \lambda'_{k-1}$, $n_h^T x^{k-1} < b_h$, $n_h^T x^k = b_h$ и $n_h^T x^\nu = b_h$ при некотором $\nu < k-1$, добавим h к Z (если $n_h = -e_j$, то $b_h = 0$; если $n_h = e_j$, то $b_h = c_j$).

Задача выбора направления примет вид

$$\max \{\sigma \mid (s, \sigma) \in S'(x^k, \varepsilon_k); \quad n_h^T s = 0, \quad h \in Z;$$
$$-(g^k)^T s + \sigma \leqslant 0; \quad \text{N1} - \text{N3 или N4}\}. \qquad (11.2.2)$$

c. Решим задачу выбора направления и получим ограниченное решение s^k, σ_k (если решение неограниченное, добавим требование ограниченности).

d. Если $\sigma_k < \varepsilon''$ и $Z \neq 0$, заменим Z на 0 и решим (11.2.1) или (11.2.2) снова.

Если $\sigma_k < \varepsilon''$, $Z = 0$ и в задаче выбора направления есть ограничения вида $(g^{l+1} - g^l)^T s = 0$, опустим их и снова решим задачу.

Если $\sigma_k < \varepsilon''$, $Z = 0$ и нет ограничений типа $(g^{l+1} - g^l)^T s = 0$ или $\sigma_k > \varepsilon''$, перейдем к шагу 2e.

e. Если $\sigma_k < \varepsilon_k$, заменим $S'(x^k, \varepsilon_k)$ в (11.2.1) или (11.2.2) на $S'\left(x^k, \frac{\varepsilon_k}{2}\right)$, заменим ε_k на $\frac{\varepsilon_k}{2}$ и снова решим задачу (11.2.1) или (11.2.2).

Если $\sigma_k = 0$ и $S'(x^k, \varepsilon_k) = S'(x^k)$, то x^k — максимальное решение.

Если $\sigma_k \geqslant \varepsilon_k$, положим $\varepsilon_{k+1} = \varepsilon_k$ и перейдем к шагу 2f.

f. Определим λ_k с помощью (7.4.1) и (7.4.4). Если $\lambda_k = \infty$, решение не ограничено.

g. Вычислим $x^{k+1} = x^k + \lambda_k s^k$, $\Delta F(x^k)$ и $F(x^{k+1})$.

h. Если $\Delta F(x^k) < \varepsilon'$, $Z = 0$ и задача выбора направления не содержит ограничений вида $(g^{l+1} - g^l)^T s = 0$, решение прекращается. Если $\Delta F(x^k) < \varepsilon'$, $Z \neq 0$ или задача выбора направления содержит ограничения вида $(g^{l+1} - g^l)^T s = 0$, в следующей задаче выбора направления заменим Z на 0 или опустим ограничения $(g^{l+1} - g^l)^T s = 0$ (т. е. действуем, как при $\lambda_k = \lambda'_k$) и повторим шаги 2a — 2h; если $\Delta F(x^k) \geqslant \varepsilon$, повторим шаги 2a — 2h.

Алгоритм P4.

1. Найдем некоторое $x^0 \in R$, выберем столь малое $\varepsilon' > 0$, что имеет смысл прекратить решение, если возрастание оптимизируемой функции окажется меньше, чем ε'.

2. Предположим, что точки $x^0, x^1, \ldots, x^k$ $(x^h \in R$, $F(x^h) > F(x^{h-1}))$ определены. Произведем следующие операции для нахождения x^{k+1}.

a. Вычислим $g^k = g(x^k)$.

b. Если $\lambda_{k-1}=\lambda''_{k-1}<\lambda'_{k-1}$ ($\lambda_{-1}=\lambda'_{-1}$, по определению), решим частную задачу линейного программирования

$$\max \{x_0 \,|\, q_i(x^k)^T x+\Theta_i x_0 \leqslant q_i(x^k)^T x^k+b_i-f_i(x^k), \quad i\in I_C;$$
$$a_i^T x \leqslant b_i, \quad i\in I_L; \quad 0\leqslant x_j\leqslant c_j, \quad j\in J;$$
$$(g^{l+1}-g^l)^T x=(g^{l+1}-g^l)^T x^k, \qquad l=r, \ldots, k-1;$$
$$-g(x^k)^T x+x_0\leqslant -g(x^k)^T x^k\}, \tag{11.2.3}$$

где r таково, что $\lambda_{r-1}=\lambda'_{r-1}$, $\lambda_h=\lambda''_h<\lambda'_h$ при $r\leqslant h\leqslant k-1$. Если $\lambda_{k-1}=\lambda'_{k-1}$, решим вспомогательную задачу линейного программирования

$$\max \{x_0 \,|\, q_i(x^k)^T x+\Theta_i x_0\leqslant q_i(x^k)^T x^k+b_i-f_i(x^k), \quad i\in I_C;$$
$$a_i^T x\leqslant b_i, \quad i\in I_L; \quad 0\leqslant x_j\leqslant c_j, \quad j\in J;$$
$$-g(x^k)^T x+x_0\leqslant -g(x^k)^T x^k\}. \tag{11.2.4}$$

c. Если решение вспомогательной задачи линейного программирования бесконечно, рассмотрим предельный луч, ведущий в бесконечность, получим s^k и перейдем к шагу 2d.

Если решение $(x^k)'$, $(x_0^k)'$ частной задачи линейного программирования конечно, $(x_0^k)'=0$ и нет ограничений типа $(g^{l+1}-g^l)^T(x-x^k)=0$, то x^k — максимальное решение. Если же в частной задаче линейного программирования имеются такие ограничения, опускаем их и решаем эту задачу вновь. Если $(x_0^k)'>0$, положим $s^k=(x^k)'-x^k$.

d. Определим λ_k с помощью (7.4.1) и (7.4.4). Если $\lambda_k=\infty$, решение не ограничено.

e. Вычислим $x^{k+1}=x^k+\lambda_k s^k$, $\Delta F(x^k)$ и $F(x^{k+1})$.

f. Если $\Delta F(x^k)<\varepsilon'$ и вспомогательная задача не содержит ограничений вида $(g^{l+1}-g^l)^T s=0$, решение прекращается. Если $\Delta F(x^k)<\varepsilon'$, но вспомогательная задача содержит ограничения подобного типа, повторим шаги 2a — 2f, предполагая, что $\lambda_k=\lambda'_k$ [так что следующая вспомогательная задача относится к типу (11.2.4)]. Если $\Delta F(x^k)\geqslant\varepsilon'$, повторим шаги 2a — 2f.

Теорема 1. *Если $F(x)$ и R удовлетворяют соответственно условиям C3 и C1* [1]) *и $\varepsilon'=0$, алгоритм P3 порождает такую последовательность точек $x^k\in R$, что $\lim F(x^k)=m=\max\{F(x)\,|\,x\in R\}$.*

Доказательство. Из условия C3 следует, что если $\lambda_k=\infty$ для некоторого k, то $F(x^k+\lambda s^k)$ не ограничена как функция λ. Если бесконечная последовательность точек x^k не ограничена либо $\sigma_k\geqslant\delta$ при фиксированном $\delta>0$ для некоторой бесконечной последовательности, то $\lim F(x)^k=\infty$, и теорема справедлива. Таким образом, при $m<\infty$ σ_k стремится к нулю, т. е. $\sigma_k<\varepsilon''$ для всех k, больших некоторого числа $k(\varepsilon'')$. Следовательно, при $k>k(\varepsilon'')$ нет разницы между алгоритмами P3 и P1, так что дальнейшее доказательство совпадает с доказательством теоремы 1 п. 7.6.

Теорема 2. *Если $F(x)$ и R удовлетворяют соответственно условиям C3 и C1* [1]) *и $\varepsilon'=0$, алгоритм P4 порождает такую последовательность точек $x^k\in R$, что $\lim F(x^k)=m=\max\{F(x)\,|\,x\in R\}$.*

Доказательство. Предположим, что $\lambda_{h-1}=\lambda''_{h-1}<\lambda'_{h-1}$ для $h\geqslant r$ и можно найти такие числа u_h, что $\sum\limits_{h\geqslant r} u_h s^h=0$. Тогда $\sum\limits_{h\geqslant r} u_h(g^{r+1}-g^r)^T s^h=0$ или $u_r(g^{r+1}-g^r)^T s^r=0$. Так как $(g^r)^T s^r>0$ и $(g^{r+1})^T s^r=0$, то $u_r=0$. Точно так же $u_{r+1}=0$, $u_{r+2}=0$ и т. д. Следовательно, векторы s^k линейно независимы, и соотношение $\lambda_{h-1}=\lambda''_{h-1}<\lambda'_{h-1}$ не может выполняться сколь угодно долго. Отсюда следует, что ограничения, связанные с принципом сопряженности, регулярно опускаются, и для некоторых k мы получаем либо $(x_0^k)'=0$, либо $\lambda_k=\lambda'_k$. Если рассматривать подпоследовательность таких k, то утверждение теоремы следует из доказательства теоремы 2 п. 7.6.

Теорема 3. *Если функция $F(x)$, удовлетворяющая условиям C3, ограничена на R и $\varepsilon'>0$, то алгоритмы P3 и P4 требуют конечного числа шагов, т. е. после конечного числа шагов $\Delta F(x^k)<\varepsilon'$, и задача выбора направления не содержит дополнительных ограничений [порожденных AZ3 (в P3) или принципом сопряженности].*

[1]) См. примечание на стр. 111. — *Прим. перев.*

[1]) См. примечание на стр. 111. — *Прим. перев.*

Доказательство. Если функция $F(x)$ ограничена, неравенство $\Delta F(x^k) \geqslant \varepsilon'$ не может иметь места сколь угодно долго. Если $\Delta F(x^k) < \varepsilon'$, а задача выбора направления содержит дополнительные ограничения, порожденные AZ3 или принципом сопряженности, то следующая задача выбора направления не содержит таких ограничений. Поэтому после конечного числа шагов мы получим $\Delta F(x^k) < \varepsilon'$, в то время как задача выбора направления не содержит дополнительных требований, порожденных AZ3 или принципом сопряженности. Таким образом, после конечного числа шагов вычисления прекращаются. Этим теорема доказана.

Возникает вопрос, как проверить близость полученного решения к оптимальной точке. Это можно сделать, линеаризуя задачу в точке x^k. При этом возникает задача линейного программирования

$$\max \{g(x^k)^T x \mid q_i(x^k)^T x \leqslant q_i(x^k)^T x^k + b_i - f_i(x^k), \quad i \in I_C;$$
$$a_i^T x \leqslant b_i, \quad i \in I_L; \quad 0 \leqslant x \leqslant c\}. \tag{11.2.5}$$

Обозначим через x' оптимальное решение этой задачи. В силу теоремы 8 п. 2.6 известно, что

$$\max \{F(x) \mid x \in R\} - F(x^k) \leqslant g(x^k)^T x' - g(x^k)^T x^k. \tag{11.2.6}$$

Следовательно, если точка $\bar{x} \in R$ считается близкой к оптимальной при $m - F(\bar{x}) < \bar{\varepsilon}$ для заданного $\bar{\varepsilon} > 0$, то точка x^k будет близка к оптимальной, когда $g(x^k)^T x' - g(x^k)^T x^k \leqslant \bar{\varepsilon}$. Эти условия достаточны, но не необходимы. Действительно, легко построить пример, в котором (11.2.5) имеет неограниченное решение для каждого k, хотя едва ли это может случиться в практической задаче (в этом случае необходимо $c_j = \infty$ для некоторых j). Однако имеет место следующая теорема.

Теорема 4. *Если последовательность x^k сходится к точке $\bar{x}$, где $F(\bar{x}) = m$, и s^k — последовательность векторов единичной длины, направленных для каждого k от x^k к решению $(x^k)'$ задачи (11.2.5) [или при неограниченном $(x^k)'$ направленных вдоль предельного луча (11.2.5), указывающего неограниченное решение], то $g(x^k)^T s^k$ стремится к нулю.*

Доказательство. Поскольку $\bar{x}$ — оптимальное решение, то можно найти такие неотрицательные числа $\bar{u}_i$, $\bar{v}_j^-$ и $\bar{v}_j^+$, что $g(\bar{x}) = \sum \bar{u}_i q_i(\bar{x}) + \sum \bar{u}_i a_i - \sum \bar{v}_j^- e_j + \sum \bar{v}_j^+ e_j$, $\bar{u}_i = 0$ при $i \notin I_C(\bar{x}) + I_L(\bar{x})$, $\bar{v}_j^- = 0$ при $\bar{x}_j > 0$ и $\bar{v}_j^+ = 0$ при $\bar{x}_j < c_j$. Для каждого k справедливо $g(x^k) = \sum \bar{u}_i q_i(x^k) + \sum \bar{u}_i a_i - \sum \bar{v}_j^- e_j + \sum \bar{v}_j^+ e_j + h(x^k)$. Из непрерывности функций $g(x^k)$ и $q_i(x^k)$ следует, что векторы $h(x^k)$ удовлетворяют условию $\lim\limits_{k\to\infty} h(x^k) = 0$, т. е. $h(x^k)^T s^k$ стремится к нулю. Для любого малого положительного ε можно найти такое $k(\varepsilon)$, что для всех $k \geqslant k(\varepsilon)$ будет выполнено неравенство $n_h^+ s^k \leqslant \varepsilon$ при $h \in H(\bar{x})$. Следовательно, $0 < g(x^k)^T s^k \leqslant \varepsilon(\sum \bar{u}_i + \sum \bar{v}_j^- + \sum \bar{v}_j^+) + h(x^k)^T s^k$. Теорема доказана.

Таким образом, если для решения x^k задачи выпуклого программирования выполняется условие $g(x^k)^T s^k \geqslant \varepsilon_1$, где s^k определено, как в теореме 4, а ε_1 — заданное число, то имеет смысл продолжать вычисления, заменяя ε', скажем, на $\varepsilon'/2$. Если $g(x^k)^T s^k < \varepsilon_1$, имеются две возможности:

1. Заранее известна величина $\bar{\lambda}_1$ — верхняя грань значений λ, удовлетворяющих условию $\bar{x} = \bar{x}^k + \lambda \bar{s}$, где $\bar{s}$ — вектор единичной длины. В этом случае положим $\varepsilon_1 = \frac{\bar{\varepsilon}}{\lambda_1}$, тогда при $g(x^k)^T s^k < \varepsilon_1$ решение близко к оптимальной точке. Если $c_j < \infty$ для всех j, можно, например, положить $\lambda_1^2 = c^T c$.

2. Нельзя заранее определить λ_1 (такая ситуация едва ли может встретиться в практических задачах). Вместо (11.2.5) решим задачу

$$\max \{g(x^k)^T x \mid q_i(x^k)^T x \leqslant q_i(x^k)^T x^k + b_i - f_i(x^k), \quad i \in I_C;$$
$$a_i^T x \leqslant b_i, \quad i \in I_L; \quad 0 \leqslant x \leqslant c;$$
$$q_i(x^l)^T x \leqslant q_i(x^l)^T x^l, \quad l = 1, 2, \ldots, k-1, i \in I_C(x^l);$$
$$-g(x^l)^T x \leqslant -g(x^l)^T x^l, \quad l = 1, 2, \ldots, k-1\}. \tag{11.2.7}$$

Все касательные гиперплоскости к гиперповерхностям $f_i(x) = b_i$ в точках x^l и все касательные гиперплоскости к гиперпо-

верхностям $F(x)=F(x^l)$ являются здесь граничными. Ясно, что $\bar{x}$ удовлетворяет всем ограничениям (11.2.7). Если $(x^k)''$ — оптимальное решение (11.2.7), тогда

при $g(x^k)^T(x^k)'' - g(x^k)^T x^k < \bar{\varepsilon}$ — решение можно прекратить,

при $g(x^k)^T(x^k)'' - g(x^k)^T x^k \geqslant \bar{\varepsilon}$ — необходимо продолжить решение.

Число строк в задаче линейного программирования (11.2.7) велико, число внебазисных переменных равно n. Для ее решения можно применить двойственный модифицированный мультипликативный алгоритм.

11.3. Вычислительные аспекты.

Можно сделать несколько замечаний, касающихся вычислительных аспектов двух алгоритмов, описанных в предыдущем пункте.

1. Если среди линейных ограничений задачи выпуклого программирования есть уравнения, можно либо предварительно исключить некоторые переменные, либо, начав с точки $x^0 \in R$, требовать $a_i^T s = 0$ вместо $a_i^T s \leqslant 0$ во всех задачах выбора направления. В первом случае число переменных сокращается, но утрачивается специальная простая структура функций $F(x)$, $f_i(x)$ и $a_i^T x$ или увеличивается число ненулевых коэффициентов в этих функциях. Среди нелинейных ограничений уравнений быть не может, так как в этом случае допустимая область не будет выпуклой (лишь в исключительных случаях область выпукла, но условия регулярности C1 не удовлетворяются).

2. Когда все ограничения линейны, введение дополнительной переменной σ излишне. В этом случае вместо σ максимизируется $g(x^k)^T s$. Алгоритм требует существенно более простых вычислений, поскольку не вычисляются векторы $q_i(x)$, т. е. упрощается переход от одной задачи выбора направления к другой; быстрее определяется и λ_k'.

3. Если $\lambda_{k-1} = \lambda_{k-1}'' < \lambda_{k-1}'$, т. е. к задаче выбора направления добавлено ограничение $(g(x^k) - g(x^{k-1}))^T s = 0$, замена соотношения $-g(x^{k-1})^T s + \sigma \leqslant 0$ на $-g(x^k)^T s + \sigma = 0$ не нужна, т. е. градиент не пересчитывается.

4. Если в алгоритме P3 для некоторого l выполняется условие $b_i - \varepsilon_k \leqslant f_i(x^k) < b_l$, так что должно быть добавлено ограничение $q_l(x^k)^T s + \Theta_l \sigma \leqslant 0$, и, если предшествующая задача выбора направления содержала ограничение $q_l(x^l)^T s + \Theta_l \sigma \leqslant 0$ для некоторого $l \leqslant k-1$ и того же l, нет нужды вычислять нормаль $q_l(x^k)$, так как можно сохранить ограничение $q_l(x^l) s + \Theta_l \sigma \leqslant 0$. Пересчет необходим лишь в случае $f_i(x^k) = b_l$. Благодаря этому число пересчитываемых строк уменьшается и применение AZ1 делается менее трудоемким.

5. Сходимость итеративного процесса может быть ускорена подходящим выбором чисел Θ_l. Эти числа могут изменяться во время вычислений. Опыт решения определенного типа задач покажет, как их следует изменить наиболее эффективно.

6. Результаты решения задачи выбора направления всегда можно использовать как исходные данные для следующей задачи. Если для решения двойственной задачи выбора направления в прямой (с N2 или N3) или двойственной форме (с N4 или N5) применяется модифицированный мультипликативный алгоритм, нетрудно использовать конечное множество η-векторов предыдущей задачи как исходное множество следующей. Однако с ростом различия между последующей и предыдущей задачами выбора направления становится выгодней начать решение следующей задачи с предварительных операций. Возможные различия последующих задач описаны в двух последних абзацах п. 6.3.

7. В случае применения алгоритма P4 можно ускорить сходимость, рассматривая следующие частные задачи линейного программирования:

$$\max \{x_0 \mid q_l(x^k)^T x + \Theta_l x_0 \leqslant q_l(x^k)^T x^k + b_l - f(x^k), \quad l \in I_C;$$
$$a_i^T x \leqslant b_i, \quad i \in I_L; \quad 0 \leqslant x \leqslant c;$$
$$-g(x^k)^T x + x_0 \leqslant -g(x^k)^T x^k;$$
$$q_l(x^l)^T x \leqslant q_l(x^l)^T x^l, \qquad l = 1, 2, \ldots, k-1, \; l \in I_C(x^l);$$
$$-g(x^l)^T x \leqslant -g(x^l)^T x^l, \quad l = 1, 2, \ldots, k-1\}. \quad (11.3.1)$$

Допустимое множество, определенное ограничениями (11.3.1), содержит множество $\{x \in R \mid F(x) \geqslant F(x^k)\}$, т. е. введение дополнительных ограничений не искажает решения. Здесь можно также добавить ограничения типа $(g(x^{l+1}) - g(x^l))^T \times \times (x - x^k) = 0$, предохраняющие от мелких шагов. Если задача типа (11.3.1) решена, нетрудно получить решение (11.2.7). Нужно только заменить все Θ_i нулями, затем определить соответствующие изменения в результатах решения (11.3.1). Все это упрощает проверку близости решения к оптимальной точке.

Заметим, наконец, что при построении вспомогательной задачи в P4 следует принимать во внимание не все ограничения $f_i(x) \leqslant b_i$, а можно ограничиться теми, для которых $b_i - \varepsilon \leqslant f_i(x^k) \leqslant b_i$ (ε — произвольное малое положительное число).

11.4. Обсуждение алгоритмов.

Мы предложили пять различных методов решения задач выпуклого программирования, соответствующих различным нормализациям. В этом пункте мы попытаемся сравнить их между собой. Будем различать случай, когда все ограничения линейны, и общий случай произвольного множества ограничений. В последнем случае определение длины шага λ_k существенно более трудоемко, поскольку нельзя получить значение λ_k' так же просто, как и в случае только линейных ограничений. Поэтому желательно располагать методом, требующим меньшего числа шагов, даже если объем работы на шаг будет бóльшим. Таким образом, иногда наиболее эффективен алгоритм P3 с нормализацией N1, особенно когда число ограничений относительно невелико, но они существенно нелинейны и содержат много ненулевых членов. Однако часто дополнительные вычисления на шаг не компенсируются уменьшением числа шагов, в основном из-за трудностей начала решения при данной нормализации. Если в задаче много ограничений с небольшим числом членов, если нелинейные ограничения в основном линейны, если оптимизируемая функция содержит мало нелинейных членов, т. е., короче говоря, если бóльшая часть задачи линейна,

тогда наиболее эффективен алгоритм P4 с нормализацией N5. Заметим также, что этот алгоритм можно модифицировать, как описано в п. 11.3 (7). Если ограничения и оптимизируемая функция существенно нелинейны, выгоднее применять алгоритм P3 с нормализациями N2, N3 или N4. В таком порядке обычно растет число шагов и убывает объем работы на шаг.

Если все ограничения линейны, а оптимизируемая функция нелинейна, можно сравнить наши методы с существующими, например разработанными Франком и Вулфом [32], Розеном [28] и Деннисом [19, гл. 7].

Теорема 1. *Алгоритм P2 в приложении к задаче с линейными ограничениями эквивалентен методу нелинейного программирования Франка и Вулфа.*

Доказательство. В методе Франка и Вулфа различаются две последовательности допустимых точек: последовательность z^k и последовательность x^k. Исходной точкой является вершина $x^0 = z^0 \in R$. На k-м шаге частная задача линейного программирования $\max\{g(x^k)^T x \mid x \in R\}$ решается симплексным методом и получается оптимальное решение x^{k+1} (или предельный луч). Затем максимизируется $F(z^k + \lambda(x^{k+1} - z^k))$ при условии, что $z^{k+1} = z^k + \lambda_k(x^{k+1} - z^k) \in R$, т. е. при условии, что $\lambda \leqslant 1$, когда вектор x^{k+1} конечен. Ясно, что этот процесс в точности эквивалентен нашему алгоритму P2, а точки z^k и x^k соответствуют точкам x^k и $(x^k)'$.

Наш алгоритм P2 решает частную задачу линейного программирования, выбирая точку z^k метода Вулфа в качестве исходной для следующей задачи и применяя к частной задаче метод возможных направлений (например, метод с нормализацией N4, являющийся прямым обобщением симплексного метода). Этот алгоритм можно применять и к задаче с нелинейными ограничениями. Его сходимость можно улучшить, используя принцип сопряженности; мы получим при этом алгоритм P4. Сходимость можно улучшить также, применяя способ, описанный в п. 11.3 (7). Как было показано в гл. 10, этот алгоритм конечен в случае квадратичной оптимизиру-

мой функции. Таким образом, его можно рассматривать как расширение метода Франка и Вулфа.

Метод проекций градиента Розена также можно рассматривать как метод возможных направлений. В этом методе строится проекция s' градиента g на конус

$$\{s \mid a_i^T s = 0, \quad i \in I(x); \quad s_j = 0, \quad j \in J^-(x) + J^+(x)\}.$$

Если $g^T s' > 0$, s' — подходящее возможное направление. Если $g^T s' = 0$, одно из уравнений, определяемое рассмотрением двойственных переменных, заменяется более слабым неравенством, что в невырожденном случае ведет к возможному вектору s_1', для которого $g^T s_1' > 0$. Длина шага определяется так же, как и в наших алгоритмах. Сходимость $F(x^k)$ к m можно доказать, если множества $I(x)$, $J^-(x)$ и $J^+(x)$, определяющие конусы, заменить на $I(x, \varepsilon)$. Вычисления, необходимые для определения вектора s', аналогичны вычислениям в P1 с нормализацией N1. Таким образом, выводы, сделанные в п. 8.6 относительно нормализации N1, применимы и к методу проекций градиента. Следует отметить только, что вектор s' в методе проекций градиента не является возможным направлением, образующим наименьший возможный угол с градиентом, поэтому можно ожидать, что число шагов в методе проекций градиента больше, чем в P1 с N1. Это особенно заметно, если из-за требований $a_i^T s = 0$ или $s_j = 0$ [вместо $a_i^T s \leqslant 0$ или $s_j \geqslant 0$ $(\leqslant 0)$] мы остаемся на некоторой гиперплоскости и делаем на ней много мелких шагов, в то время как эта гиперплоскость не содержит оптимальной точки.

Метод Денниса полностью эквивалентен нашему алгоритму P1 с N1. Однако Деннис не рассматривает задачи, содержащие нелинейные ограничения и не приводит эффективной вычислительной схемы для решения квадратичной задачи выбора направления типа (8.2.1).

Мультиплексный метод Фриша [33] является еще одним методом, использующим принципы, сравнимые с P1 при нормализации N1. Этот метод был разработан сначала для задачи линейного программирования. Задача проектирования градиента на конус возможных направлений в этом методе вообще не рассматривается.

ЛИТЕРАТУРА

1. Беллман Р., Динамическое программирование, ИЛ, М., 1960.
2. Бендер (J. F. Benders) Partitioning in Mathematical Programming, Thesis, Utrecht, 1960.
3. Бил (E. M. L. Beale) An Alternative Method of Linear Programming, *Proc. Cambr. Phil. Soc.*, **50** (1954), 513—523.
4. Бил (E. M. L. Beale) On Minimizing a Convex Function Subject to Linear Inequalities, *J. Roy. Statist. Soc.* (B), **17** (1955), 173—184.
5. Бил (E. M. L. Beale) On Quadratic Programming, *Nav. Res. Log. Quart.*, **6** (1959), 227—243.
6. Вагнер (H. M. Wagner) A Comparison of the Original and Revised Simplex Method, *J. Oper. Res. Soc. Am.*, **5** (1957), 361—369.
7. Вагнер (H. M. Wagner) The Dual Simplex Algorithm for Bounded Variables, *Nav. Res. Log. Quart.*, **5** (1958), 257—261.
8. Вайда С. Теория игр и линейное программирование, Сб. Линейные неравенства и смежные вопросы, ИЛ, М., 1959.
9. Вулф (P. Wolfe) The Simplex Method for Quadratic Programming, *Econometrica* **27** (1959), 382—398.
10. Гасс С., Линейное программирование, Физматгиз, М., 1961.
11. Гасс, Саати (S. I. Gass, T. Saaty) The Computational Algorithm for the Parametric Objective Function, *Nav. Res. Log. Quart.*, **2** (1955), 39—45.
12. Гуд (R. A. Good) Systems of Linear Relations, *Soc. Ind. Appl. Math. Review*, **1** (1959), 1—31.

13. Данциг (G. B. Dantzig)
Computational Algorithm of the Revised Simplex Method, The Rand Corporation, Santa Monica, Cal. RM-1266, 26 Octobor 1953.

14. Данциг (G. B. Dantzig)
Upper Bounds, Secondary Constraints and Block Triangularity in Linear Programming, *Econometrica*, **23** (1955), 174—183.

15. Данциг Г., Форд Л., Фулкерсон Д., Алгорифм для одновременного решения прямой и двойственной задач линейного программирования, Сб. Линейные неравенства и смежные вопросы, ИЛ, М., 1959.

16. Данциг, Орчард-Хейс (G. B. Dantzig, Wm. Orchard Hays)
Alternate Algorithm for the Revised Simplex Method Using Product Form for the Inverse, The Rand Corporation, Santa Monica, Cal., RM-1268, 19 November 1953.

17. Данциг, Орден, Вулф (G. B. Dantzig, A. Orden, P. Wolfe)
The Generalized Simplex Method for Minimizing a Linear Form under Linear Inequality Constraints, *Pacif. J. Math.*, **5** (1955), 183—195.

18. Данциг, Вулф (G. B. Dantzig, P. Wolfe)
Decomposition Principle for Linear Programs, *J. Oper. Res. Soc. Am.*, **8** (1960), 101—111.

19. Деннис Дж. Б.
Математическое программирование и электрические цепи, ИЛ, М., 1961.

20. Зойтендейк (G. Zoutendijk)
Maximizing a Function in a Convex Region, *J. Roy. Statist. Soc.* (B), **21** (1959), 338—355.

21. Йовелл (W. Jewell)
Optimal Flow through Networks, Interim Technical Report № 8, June 1958, Massachusetts Institute of Technology.

22. Кун, Таккер (H. W. Kuhn, A. W. Tucker)
Non-Linear Programming, in: «Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability», Berkeley, Cal., 1950, 481—492.

23. Лемке (C. E. Lemke)
The Dual Method of Solving the Linear Programming Problem, *Nav. Res. Log. Quart.*, **1** (1954), 36—47.

24. Марковиц (H. M. Markowitz)
The Optimization of a Quadratic Function Subject to Linear Constraints, *Nav. Res. Log. Quart.*, **3** (1956), 111—133.

25. Марковиц (H. M. Markowitz)
The Elimination Form of the Inverse and Its Application to Linear Programming, *Management Science*, **3** (1957), 255—269.

26. Орчард-Хейс (Wm. Orchard-Hays)
Evolution of Computer Codes for Linear Programming, The Rand Corporation, Santa Monica, Cal., P-810, 14 March 1956.

27. Рилей, Гасс (V. Riley, S. I. Gass)
Linear Programming and Associated Techniques, The John Hopkins Press, Baltimore, Md., 1958.

28. Розен (J. B. Rosen)
The Gradient Projection Method for Nonlinear Programming, Part I — Linear Constraints, *J. Soc. Indust. and Appl. Maths*, **8** (1960), 181—217.

29. Таккер А.
Двойственные системы однородных линейных соотношений, Сб. Линейные неравенства и смежные вопросы, ИЛ, М., 1959.

30. Фаркаш (J. Farkas)
Theorie der einfachen Ungleichungen, *J. reine und angew. Math.*, **124** (1902), 1—27.

31. Форд, Фулкерсон (L. Ford Jr., D. R. Fulkerson)
A Primal — Dual Algorithm for the Capacitated Hitchcock Problem, *Nav. Res. Log. Quart.*, **4** (1957), 47—54.

32. Франк, Вулф (M. Frank, P. Wolfe)
An Algorithm for Quadratic Programming, *Nav. Res. Log. Quart.*, **3** (1956), 95—110.

33. Фриш (R. A. K. Frisch)
The Multiplex Method for Linear Programming, Memo. Univ. Inst. of Economics, Oslo, 17 October 1955.

34. Хестенс, Штифель (M. R. Hestenes, E. Stiefel)
Methods of Conjugate Gradient for Solving Linear Sistems, *J. Res. Nat. Bur. Standards*, **49** (1952), 409—436.

35. Хилдрет (C. Hildreth)
A Quadratic Programming Procedure, *Nav. Res. Log. Quart.*, **4** (1957), 79—85.

36. Чарнс (A. Charnes)
Optimality and Degeneracy in Linear Programming, *Econometrica*, **20** (1952), 160—170.

37. Чарнс, Лемке (A. Charnes, C. E. Lemke) Computational Theory of Linear Programming, Part I. The «Bouned Variables» Problem. Office of Naval Research, Research Memorandum № 10, Carnegie Institute of Technology, 7 January 1954.

38. Чарнс, Лемке (A. Charnes, C. E. Lemke) Minimization of Non-Linear Separable Convex Function, *Nav. Res. Log. Quart.*, 1 (1954), 301—312.

39. Чени, Голдстайн (E. W. Cheney, A. A. Goldstein) Newton's Method for Convex Programming and Tchebycheff Approximation, *Numerishe Mathematik*, 1 (1959), 253—268.

40. Эрроу, Гурвиц, Удзава, Исследования по линейному и нелинейному программированию, ИЛ, М., 1962.

41*. Юдин Д. Б., Гольштейн Е. Г. Задачи и методы линейного программирования, Физматгиз, М., 1961.

ОГЛАВЛЕНИЕ

Предисловие редактора перевода 5
Глава 1. **Введение** 7
1.1. История 7
1.2. Современное состояние математического программирования 8
1.3. Предмет монографии 11
1.4. Обозначения 14
Глава 2. **Теория выпуклого программирования** 16
2.1. Введение 16
2.2. Теория линейных неравенств 16
2.3. Определенные матрицы 19
2.4. Выпуклые функции и области 23
2.5. Задача линейного программирования 29
2.6. Задача выпуклого программирования 34
Глава 3. **Методы решения задач линейного программирования** 41
3.1. Введение 41
3.2. Симплексный метод 41
3.3. Двойственный симплексный метод 48
3.4. Метод одновременного решения прямой и двойственной задач 51
Глава 4. **Вычислительные алгоритмы симплексного и двойственного симплексного методов** 54
4.1. Введение 54
4.2. Прямой алгоритм 55
4.3. Алгоритм с обратной матрицей 56
4.4. Мультипликативный алгоритм 59
4.5. Модифицированный мультипликативный алгоритм 62
4.6. Вычислительные алгоритмы двойственного симплексного метода 67
Глава 5. **Численное сравнение различных алгоритмов** 70
5.1. Введение 70
5.2. Критерии сравнения 70
5.3. Поведение практических задач линейного программирования 72
5.4. Предположения о поведении задач линейного программирования 73
5.5. Сравнение алгоритмов 77
Заключение 81
Глава 6. **Некоторые специальные задачи линейного программирования** 82
6.1. Введение 82
6.2. Задачи с двусторонними ограничениями на переменные 82
6.3. Задача с абсолютными значениями 86

Глава 7. **Методы возможных направлений** 94
7.1. Введение 94
7.2. Определение исходного возможного решения 96
7.3. Определение подходящего возможного направления . 99
7.4. Определение длины шага 103
7.5. Алгоритмы 104
7.6. Сходимость последовательности решений 108
7.7. Нелинейное программирование без предположений о выпуклости 114

Глава 8. **Нормализации возможных направлений** 117
8.1. Введение 117
8.2. Нормализация N1 117
8.3. Нормализация N2 131
8.4. Нормализация N3 132
8.5. Другие нормализации 133
8.6. Сравнение различных нормализаций 136

Глава 9. **Задача линейного программирования и методы возможных направлений** 138
9.1. Введение 138
9.2. Обзор алгоритмов 138
9.3. Вычислительные аспекты 141
9.4. Обсуждение алгоритмов 142
9.5. Задачи линейного программирования больших размеров 147

Глава 10. **Квадратичное программирование** 150
10.1. Введение 150
10.2. Описание алгоритмов 150
10.3. Обсуждение методов квадратичного программирования 156

Глава 11. **Выпуклое программирование** 159
11.1. Введение 159
11.2. Алгоритмы выпуклого программирования 159
11.3. Вычислительные аспекты 166
11.4. Обсуждение алгоритмов 168

Литература 171

Г. Зойтендейк

МЕТОДЫ ВОЗМОЖНЫХ НАПРАВЛЕНИЙ

Редактор *А. А. Бряндинская*
Художественный редактор *В. И. Шаповалов*
Технический редактор *С. В. Приданцева*
Корректор *И. С. Цветкова*

Сдано в производство 2/VIII 1962 г. Подписано к печати 28/XI 1962 г. Бумага 84×108$^1/_{32}$=2,8 бум. л. 9,0 печ. л. Уч.-изд. 8,5 л. Изд. № 1/0901. Цена 60 коп. Зак. № 613.

ИЗДАТЕЛЬСТВО ИНОСТРАННОЙ ЛИТЕРАТУРЫ.
Москва, 1-й Рижский пер., 2

Типография № 2 им. Евг. Соколовой УЦБ и ПП Ленсовнархоза.
Ленинград, Измайловский пр., 29.